PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2006-107514
(43)Date of publication of application : 20.04.2006

(51)Int.Cl. *G06F 15/80 (2006.01)*
G06F 9/52 (2006.01)
G06F 9/54 (2006.01)

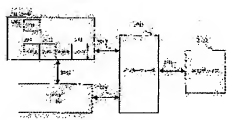
(21)Application number : 2005-292708
(22)Date of filing : 05.10.2005

(71)Applicant : SONY COMPUTER ENTERTAINMENT INC
(72)Inventor : SUZUOKI MASAKAZU
YAMAZAKI TAKESHI

(30)Priority
Priority number : 2004 959635 Priority date : 05.10.2004 Priority country : US

(54) SYSTEM AND DEVICE WHICH HAVE INTERFACE DEVICE WHICH CAN PERFORM DATA COMMUNICATION WITH EXTERNAL DEVICE

(57)Abstract:
PROBLEM TO BE SOLVED: To provide an architecture which enables various members on a network to share data and applications among them without an additional load in computing.
SOLUTION: A system is provided, which has a processing element (PE), an input/output (I/O) interface device, and a shared memory. The PE has at least one processing unit (PU) and also has one or more additional processing devices (APU). At least one of the APUs executes an I/O function by reading from or writing to an external device connected to the I/O interface device 2915. Data is exchanged between the APU and the I/O interface device 2915 by using a data level synchronization mechanism through the shared memory.



(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2006-107514
(P2006-107514A)
(43)公開日　平成18年4月20日(2006.4.20)

(51)Int.Cl.		F I		テーマコード（参考）
G06F 15/80	(2006.01)	G06F 15/80		
G06F 9/52	(2006.01)	G06F 9/46	475C	
G06F 9/54	(2006.01)	G06F 9/46	480Z	

審査請求 有　請求項の数 27　OL　（全 50 頁）

(21)出願番号　特願2005-292708(P2005-292708)
(22)出願日　平成17年10月5日(2005.10.5)
(31)優先権主張番号　10/959,635
(32)優先日　平成16年10月5日(2004.10.5)
(33)優先権主張国　米国(US)

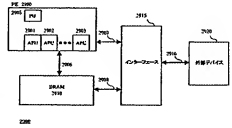
（特許庁注：以下のものは登録商標）
1．ＪＡＶＡ
2．イーサネット

(71)出願人　395015319
株式会社ソニー・コンピュータエンタテインメント
東京都港区南青山二丁目６番２１号
(74)代理人　100099324
弁理士　鈴木　正剛
(74)代理人　100108604
弁理士　村松　義人
(74)代理人　100111615
弁理士　佐野　良太
(72)発明者　鈴置　雅一
東京都港区南青山二丁目６番２１号　株式会社ソニー・コンピュータエンタテインメント内

最終頁に続く

(54)【発明の名称】外部デバイスとデータ通信可能なインターフェイスデバイスを有するシステム及び装置

(57)【要約】
【課題】計算上の負担が付加されることなく、ネットワークの様々なメンバー間でのデータとアプリケーションの共用可能なアーキテクチャを提供する。
【解決手段】処理エレメント（ＰＥ）、入出力（Ｉ／Ｏ）インターフェースデバイス及び共有メモリを有するシステムが提供される。ＰＥは、処理ユニット（ＰＵ）を少なくとも一つ備え、かつ、一つ以上の付加処理装置（ＡＰＵ）を備える。ＡＰＵの少なくとも一つは、Ｉ／Ｏインターフェースデバイス２９１５に接続された外部デバイスに対して読み出しあるいは書き込みを行うことでＩ／Ｏ機能を実行する。データは、ＡＰＵとＩ／Ｏインターフェースデバイス２９１５との間で、データレベル同期メカニズムを用いて共有メモリを通じて交換される。
【選択図】図２９



【特許請求の範囲】
【請求項１】
外部デバイスへのデータ通信と外部デバイスからのデータ通信とを行うように動作可能なインターフェースデバイスと、
各々が前記データの格納をするように動作可能な、複数のメモリ・ロケーションを持つメモリとを含み、
前記インターフェースデバイスと前記メモリのうちの少なくとも１つが、前記メモリ・ロケーションの対応するメモリ・ロケーションと関連付けられた状態情報を格納するように動作可能であって、前記状態情報には、第１フィールドとアドレス・フィールドとが含まれ、この第１フィールドとアドレス・フィールドとは、与えられたメモリ・ロケーションに対して、前記関連する状態情報の前記第１フィールドの値が第１の値と等しくかつ前記関連する状態情報の前記アドレス・フィールドの値が第２の値と等しい場合に、前記メモリ・ロケーションへの書き込みオペレーションによって、前記メモリ・ロケーションに現在格納されているデータが、前記第２の値によって示されるアドレスへの書き込みが行われる、システム。
【請求項２】
前記アドレスは前記外部デバイスと関連付けられている、請求項１記載のシステム。
【請求項３】
前記アドレスは、前記インターフェースと通信を行うプロセッサと関連付けられているメモリ・ロケーションと関連付けられている、請求項１記載のシステム。
【請求項４】
データアクセス用に前記メモリに結合され、かつ前記インターフェースと結合されているプロセッサを含み、前記プロセッサは前記外部デバイスと関連付けられた要求を発行するように動作可能である、請求項１又は２又は３記載のシステム。
【請求項５】
少なくとも１つの前記要求は、前記外部デバイスからのデータの読み出しに対するものである、請求項４記載のシステム。
【請求項６】
少なくとも１つの前記要求は、前記外部デバイスへのデータの書き込みに対するものである、請求項４記載のシステム。
【請求項７】
前記メモリは、前記プロセッサと前記インターフェースデバイスとの間に前記要求を送るように動作可能である、請求項４又は５又は６記載のシステム。
【請求項８】
前記外部デバイスは、前記インターフェースデバイスと結合されている、請求項１～７のいずれかに記載のシステム。
【請求項９】
前記インターフェースデバイスは、ＤＭＡ伝送を用いて、前記外部デバイスと前記メモリとの間でのデータ通信を行う、請求項１～８のいずれかに記載のシステム。
【請求項１０】
複数のメモリ・ロケーションを持つメモリと、
少なくとも１つの第１のタイプと第２のタイプの要求を発行するように動作可能で前記メモリと結合されたプロセッサと、
インターフェースデバイスと、を含み、前記インターフェースデバイスは、前記第１のタイプの要求を受信したことに応答して、前記外部デバイスから前記メモリへデータを伝送し、前記メモリへ前記データを格納するように動作可能であり、かつ、前記インターフェースデバイスは前記第２のタイプの要求を受信したことに応答して、前記メモリのストレージから前記外部デバイスへデータを伝送するように動作可能であって、
前記メモリの少なくとも１つのメモリ・ロケーションと、前記インターフェースデバイスとは、ブロッキング状態を含む、複数の異なるメモリの状態をサポートし、このサポー

トでは、前記ブロッキング状態にある与えられたメモリ・ロケーションへの書き込みオペレーションによって、現在、その中に格納されているデータが、前記与えられたメモリ・ロケーションに関連付けられたアドレスに書き込まれる、システム。

【請求項１１】

前記アドレス値は前記外部デバイスを表す、請求項１０記載のシステム。

【請求項１２】

前記プロセッサと関連付けられたローカル・メモリを更に含み、前記アドレス値は前記ローカル・メモリのメモリ・ロケーションを表わす、請求項１０又は１１記載のシステム。

【請求項１３】

前記第１のタイプの前記要求は、前記外部デバイスからのデータの読み出しに対するものである、請求項１０又は１１又は１２記載のシステム。

【請求項１４】

前記第２のタイプの前記要求は、前記外部デバイスからのデータの書き込みに対するものである、請求項１０又は１１又は１２記載のシステム。

【請求項１５】

前記外部デバイスは前記インターフェースデバイスと結合されている、請求項１０～１４のいずれかに記載のシステム。

【請求項１６】

前記インターフェースデバイスは、ＤＭＡ伝送を用いて、前記外部デバイスと前記メモリとの間でのデータ通信を行う、請求項１０～１５のいずれかに記載のシステム。

【請求項１７】

前記プロセッサは、プロセシングユニット（ＰＵ）と、前記要求を発行するように動作可能な、少なくとも１つの付加処理ユニット（ＡＰＵ）とを含む、請求項１０～１６のいずれかに記載のシステム。

【請求項１８】

前記インターフェースデバイスは、前記メモリ・ロケーションへのアクセスを制御するための保護テーブルを含む、請求項１７記載のシステム。

【請求項１９】

前記ＰＵは、前記保護テーブルの値を初期化する、請求項１８記載のシステム。

【請求項２０】

外部装置とプロセッサ間を相互接続する装置であって、

保護テーブルを含み、前記保護テーブルは前記プロセッサによってアクセス可能な共用メモリの領域を示す情報を格納するように動作可能であって、

伝送コントローラを含み、前記伝送コントローラはプロセッサからの要求に応答して、前記外部デバイスと前記共用メモリの前記アクセス可能な領域のうちの少なくとも１つとの間のデータ伝送を制御するものであって、かつ、

伝送バスを含み、前記伝送バスは前記伝送コントローラによる制御の下で、前記外部デバイスと、前記共用メモリの前記少なくとも１つのアクセス可能な領域との間に前記データを伝送するように動作可能であり、

前記伝送コントローラは、メモリ・ロケーションの最新のメモリ状態によって、前記少なくとも１つのアクセス可能な領域のメモリ・ロケーションと外部デバイスとの間のデータ伝送を同期して制御するよう動作可能であり、この制御では、最新のメモリ状態がブロッキング状態である場合、メモリ・ロケーションへの書き込みオペレーションにより、現在メモリ・ロケーションに格納されているデータが、前記メモリ・ロケーションに関連付けられたアドレスへ書き込まれる、装置。

【請求項２１】

インターフェースデバイスを介して、プロセッサと外部デバイスとの間にデータ伝送を行う方法であって、

前記外部デバイスと前記プロセッサに対してローカルであるメモリとの間で、前記デー

タを、複数のメモリ・ロケーションを持つ共用メモリを介して前記プロセッサへ伝送し、前記共用メモリは、ブロッキング状態を含む複数の異なるメモリ状態をサポートし、前記ブロッキング状態にある与えられたメモリ・ロケーションへの書き込みオペレーションにより、現在前記メモリ・ロケーションに格納されているデータが、前記所望のメモリ・ロケーションと関連付けられたアドレス値へ書き込まれる、方法。

【請求項２２】

前記プロセッサは第１プロセッサであり、更に、

第２プロセッサからインターフェースデバイスへ、インターフェースデバイスの保護テーブルにおいて用いるための値を送り、前記値は前記共用メモリの一部へのアクセスを制御する、請求項２１記載の方法。

【請求項２３】

前記外部デバイスは、

外部デバイスを通じて、伝送されるデータを要求するコマンドを受信し、

前記コマンドに関連付けられたアドレス値が有効であるかどうかを判断するために、保護テーブルの値をチェックし、

アドレス値が有効である場合は、前記初期化ステップを行う、請求項２１又は２２記載の方法。

【請求項２４】

前記関連するアドレス値は前記外部デバイスを示す、請求項２１又は２２又は２３記載の方法。

【請求項２５】

前記関連するアドレス値は前記プロセッサと関連付けられたローカル・メモリを示す、請求項２１又は２２又は２３記載の方法。

【請求項２６】

デバイスであって、

要求や応答を複数の要求／応答チャネルを介して転送するための第１バスを含み、各要求／応答チャネルはプロセッサと関連付けられており、各要求／応答チャネルは、前記関連付けられたプロセッサと外部デバイスとの間にデータ通信用の要求を転送するものであり、

メモリとのデータ通信を行う第２バスを含み、前記第２バスはデータレベル同期を用いて前記データの通信時に用いられるものであり、前記メモリは複数のメモリ・ロケーションを含み、前記各メモリ・ロケーションは、ブロッキング状態を含む複数の異なるメモリ状態をサポートするものであって、前記ブロッキング状態にある与えられたメモリ・ロケーションへのデータの書き込みオペレーションにより、現在前記与えられたメモリ・ロケーションに格納されているデータが、前記与えられたメモリ・ロケーションと関連付けられているアドレス値へ書き込まれる、デバイス。

【請求項２７】

前記第１バスは、前記プロセッサによりアクセス可能なメモリの一部を示す前記値を、保護テーブルで用いられるように転送する、請求項２６記載のデバイス。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、同時係属中であり、本発明の譲受人に譲渡された以下の米国特許出願、２００１年３月２２日出願の米国特許出願第０９／８１６，００４号「広帯域ネットワークのコンピュータ・アーキテクチャおよびソフトウエア・セル」、２００１年３月２２日出願の米国特許出願第０９／８１５，５５４号「広帯域ネットワーク用のコンピュータ・アーキテクチャのデータ同期システムおよび方法」、２００１年３月２２日出願の米国特許出願０９／８１６，０２０号「広帯域ネットワーク用のコンピュータ・アーキテクチャのメモリ保護システムおよび方法」、２００１年３月２２日出願の米国特許出願第０９／８１５，５５８号「広帯域ネットワーク用のコンピュータ・アーキテクチャの資源占有システ

ムおよび方法」、および２００１年３月２２日出願の米国特許出願第０９／８１６，７５２号「広帯域ネットワーク用のコンピュータ・アーキテクチャの処理モジュール」の一部継続出願であり、その全てをここに援用する。

【背景技術】

【０００２】

本発明はコンピュータ・プロセッサ用アーキテクチャとコンピュータ・ネットワークに関し、より詳細には広帯域環境におけるコンピュータ・プロセッサ及びコンピュータ・ネットワーク用アーキテクチャに関する。

【０００３】

コンピュータ及び現今のコンピュータ・ネットワークのコンピューティング・デバイス（オフィスのネットワークで使用されるローカル・エリア・ネットワーク（ＬＡＮ）やインターネットなどのようなグローバルネットワークなど）の計算用デバイスは、スタンド・アローン型の計算用として主として設計されていた。コンピュータ・ネットワークを介するデータとアプリケーション・プログラム（“アプリケーション”）の共用は、これらのコンピュータ及びコンピューティング・デバイスの主要な設計目標ではなかった。これらのコンピュータとコンピューティング・デバイスは、様々な異なるメーカー（モトローラ、インテル、テキサス・インスツルメント、ソニーなど）により製造された広範囲の異なるタイプのプロセッサを用いて一般に設計されたものである。これらのプロセッサの各々はそれ自身の特定の命令セットと命令セット・アーキテクチャ（ＩＳＡ：instruction set architecture）とを有している。すなわち、それ自身の特定のセットのアセンブリ言語命令と、これらの命令を実行する主演算デバイスと記憶デバイスのための構造とを有する。従って、プログラマは各プロセッサの命令セットとＩＳＡとを理解してこれらのプロセッサ用のアプリケーションを書くことを要求される。今日のコンピュータ・ネットワーク上でのコンピュータとコンピューティング・デバイスに異なった種類が混在していることから、データとアプリケーションの共用及びその処理は複雑になっている。さらに、この複数種が混在する環境に対する調整を行うために、多くの場合、同じアプリケーションであっても複数のバージョンを用意することが必要となっている。

【発明の開示】

【発明が解決しようとする課題】

【０００４】

グローバルネットワーク、特にインターネットに接続されたタイプのコンピュータやコンピューティング・デバイスは広範囲に及ぶ。パーソナルコンピュータ（ＰＣ）やサーバーに加えて、これらのコンピューティング・デバイスの中には携帯電話、移動用コンピュータ、個人用情報機器（ＰＤＡ：personal digital assistant）、セット・トップ・ボックス、デジタルテレビ並びにその他のデバイスが含まれる。コンピュータやコンピューティング・デバイスにおいて異種製品が混在する中でのデータやアプリケーションを共用することに起因して、重要な問題が発生している。

【０００５】

これらの問題を解決するためのいくつかの手法が試みられてきた。これらの手法の中には特に、優れたインターフェースと複雑なプログラミング手法が含まれる。多くの場合、これらの解決方法では、処理パワーの実質的増加の実現が要求される。また、これらの解決方法では、多くの場合、アプリケーションの処理に必要な時間と、ネットワークを介するデータ伝送に必要な時間とが実質的に増加してしまうという結果が生じる。

【０００６】

一般に、データは対応のアプリケーションとは別々に、インターネットを介して伝送される。この手法では、アプリケーションに対応した各セットの伝送データにアプリケーション自体をも送る必要はなくなっている。従ってこの手法により、必要とされる帯域幅の量は最小化されるものの、ユーザには不満の原因となることも多々ある。つまり、クライアント側のコンピュータでは、この伝送データを利用するための適正なアプリケーション、あるいは最新のアプリケーションを入手できない事態も生じうる。またこの手法では、

ネットワーク上のプロセッサによって用いられている複数の異種ＩＳＡと命令セットに対応して、各アプリケーション毎にバージョンの異なる複数のアプリケーションを用意することが要求される。

【０００７】

Ｊａｖａモデルではこの問題の解決が試みられている。このモデルでは厳しいセキュリティ・プロトコルに準拠する小さなアプリケーション（“アプレット”（applet））が用いられている。アプレットはネットワークを介してサーバー側コンピュータから送信され、ククライアント側コンピュータ（“クライアント”）により実行される。異なるＩＳＡを使用しているクライアントごとに、同じアプレットであっても異なるバージョンを送信するという事態を避ける必要があるため、全てのＪａｖａアプレットはクライアント側のＪａｖａ仮想マシン上で実行される。Ｊａｖａ仮想マシンとは、ＪａｖａＩＳＡとＪａｖａ命令セットを持つコンピュータをエミュレートするソフトウエアである。しかしながらこのソフトウエアはクライアント側のＩＳＡとクライアント側の命令セットにより実行される。クライアント側ではＩＳＡと命令セットが各々異なるが、与えられるＪａｖａ仮想マシンのバージョンは1つである。従って、複数の各アプレットに異なるバージョンを用意する必要はない。各クライアントでは、当該クライアントにおけるＩＳＡと命令セットに対応した適切なＪａｖａ仮想マシンだけをダウンロードすれば、全てのＪａｖａアプレットを実行できる。

【０００８】

各々の異なるＩＳＡ命令セットに対して異なるバージョンのアプリケーションを書かなければならないという課題は解決されているものの、Ｊａｖａの処理モデルでは、クライアント側のコンピュータに対してソフトウエアの追加層が要求される。ソフトウエアのこの追加層のためにプロセッサの処理速度は著しく低下する。この速度の低下は、リアルタイムのマルチメディア・アプリケーションについて特に著しい。また、ダウンロードされたＪａｖａアプレットの中には、ウイルス、処理上の誤動作などが含まれている可能性がある。これらのウイルスと誤動作は、クライアントのデータベースの破損やその他の損害の原因となる可能性がある。Ｊａｖａモデルで用いられているセキュリティ用プロトコルでは、“サンドボックス”（Ｊａｖａアプレットがそれ以上はデータを書き込むことができない、クライアント側のメモリ内のスペース）というソフトウエアを設けることによりこの問題の解決が試みられているとはいえ、このソフトウエア駆動型セキュリティ・モデルは多くの場合、その実行時に不安定な状態になり、より多くの処理を必要とする。

【０００９】

リアルタイムのマルチメディア・ネットワーク用アプリケーションがますます重要なものになりつつある。これらのネットワーク用アプリケーションは非常に高速な処理が要求される。将来、そのようなアプリケーション用として、毎秒何千メガビットものデータが必要になるかもしれない。ネットワークの現今のアーキテクチャ、及び、特にインターネットのアーキテクチャ、並びにＪａｖａモデルなどで現在実施されているプログラミング・モデルでこのような処理速度に到達することは非常に難しい。

【００１０】

従って、新しいコンピュータ・アーキテクチャと、コンピュータ・ネットワーク用の新しいアーキテクチャと、新しいプログラミング・モデルとが求められている。この新しいアーキテクチャとプログラミング・モデルによって計算上の負担が付加されることなく、ネットワークの様々なメンバー間でのデータとアプリケーションの共用という問題が解決されることが望ましい。この新たなコンピュータ・アーキテクチャとプログラミング・モデルとにより、ネットワークのメンバー間でのアプリケーションとデータの共用時に生じるセキュリティ上の問題も解決されることが望ましい。

【課題を解決するための手段】

【００１１】

広帯域ネットワークを介して高速処理を行うコンピュータ・アーキテクチャが提供される。特に、及び、本発明によれば、システムにはデータ通信用のインターフェースデバイ

スと、データ格納用のメモリとが含まれ、メモリには少なくとも１つのメモリ・ロケーションと、それに関連する状態情報とが含まれる。状態情報には第１フィールドとアドレス・フィールドとが含まれ、第１フィールド値が第１の値に等しく、また、アドレス・フィールド値が第２の値に等しい場合、続いて行われる、メモリ・ロケーションに関連付けられたデータの書き込みによって、その中に格納されているデータが、アドレス・フィールド値により示されるアドレスへ書き込まれることになる。

【発明を実施するための最良の形態】

【００１２】

本発明の一実施形態においては、システム構成にはプロセッサ・エレメント（ＰＥ）と、入力／出力（Ｉ／Ｏ）インターフェースデバイスと、共用メモリとが含まれる。ＰＥにはさらに、少なくとも１つの処理ユニット（ＰＵ）と、１つ以上の付加処理ユニット（ＡＰＵ：attached processing unit）とが含まれる。少なくとも１つのＡＰＵは、Ｉ／Ｏインターフェースデバイスと結合されている外部デバイスからのデータの読出しと外部デバイスへのデータの書き込みを行うことにより、Ｉ／Ｏ機能を行う。データはＡＰＵとＩ／Ｏインターフェースデバイスとの間で、共用メモリを介して、データレベル同期機構を用いて交換される。特に、共用メモリにはデータ格納用の少なくとも1つのメモリ・ロケーションと関連付けられた、少なくとも１つの状態情報ロケーションが含まれ、この状態情報ロケーションには第１フィールドとアドレス・フィールドが含まれる。第１フィールド値が第１の値に等しく、また、アドレス・フィールド値が第２の値に等しい場合、続いて行われる、メモリ・ロケーションに関連付けられたデータの書き込みによって、その中に格納されているデータが、アドレス・フィールド値により示されるアドレスへ書き込まれることになる。

【００１３】

図１に、本発明によるコンピュータ・システム１０１のアーキテクチャ全体を示す。

【００１４】

この図に例示されているように、システム１０１にはネットワーク１０４が含まれ、複数のコンピュータとコンピューティング・デバイスがこのネットワークと接続されている。ネットワーク１０４の例として、ＬＡＮ、インターネットなどのグローバルネットワーク、又は他のコンピュータ・ネットワークが挙げられる。

【００１５】

ネットワーク１０４に接続されているコンピュータとコンピューティング・デバイス（ネットワークの“メンバー”）の中には、クライアント側コンピュータ１０６、サーバー側コンピュータ１０８、個人用情報機器（ＰＤＡ：personal digital assistant）、デジタルテレビ（ＤＴＶ：digital television）１１２、及びその他の有線又は無線コンピュータとコンピューティング・デバイスなどが含まれる。ネットワーク１０４のメンバーにより用いられるプロセッサは、同じ共通のコンピューティング・モジュールから構成される。またこれらのプロセッサは、好適には、ＩＳＡが全て同じで、同じ命令セットに従って処理を実行する。個々のプロセッサ内に含まれるモジュールの数は、そのプロセッサが必要とする処理パワーにより決められる。

【００１６】

例えば、システム１０１のサーバー１０８はクライアント１０６より多いデータ及びアプリケーション処理を実行するので、サーバー１０８はクライアント１０６よりも多いコンピューティング・モジュールを含むことになる。一方、ＰＤＡ１１０では、最低量の処理しか実行されない。従って、ＰＤＡ１１０には最小の数のコンピューティング・モジュールしか含まれない。ＤＴＶ１１２はクライアント１０６とサーバー１０８間の処理レベルを実行する。従ってＤＴＶ１１２にはクライアント１０６とサーバー１０８の間のいくつかのコンピューティング・モジュールが含まれる。以下に解説するように、各コンピューティング・モジュールの中には、処理用コントローラと、ネットワーク１０４を介して伝送されるデータ及びアプリケーションの並列処理を実行する複数の同一処理ユニットとが含まれる。

【００１７】
システム１０１がこのように均質な構成を有することから、アダプタビリティ、処理速度、及び処理効率が改善される。システム１０１の各メンバーが、同じコンピューティング・モジュールの１つ以上（又はコンピューティング・モジュールの一部）を用いて処理を実行するので、データ及びアプリケーションの実際の処理をどのコンピュータ又はコンピューティング・デバイスで実行するかは重要ではなくなる。更に、個々のアプリケーション及びデータの処理は、ネットワークのメンバー間で分担することができる。システム全体を通じて、システム１０１が処理したデータ及びアプリケーションを含むセルを一意的に識別することにより、この処理がどこで行われたかに関わらず、処理を要求したコンピュータ又はコンピューティング・デバイスへその処理結果を伝送することが可能になる。この処理を実行するモジュールが共通の構造と共通のＩＳＡとを有するので、プロセッサ間の互換性を達成するためのソフトウエアの追加層の計算上の負荷が回避される。このアーキテクチャとプログラミング・モデルにより、リアルタイムのマルチメディア・アプリケーションなどの実行に必要な処理速度が改善される。
【００１８】
システム１０１により改善された処理速度と効率という利点を更に利用するために、このシステムにより処理されるデータ及びアプリケーションは、一意的に識別され、それぞれフォーマットが同じであるソフトウエア・セル１０２へとパッケージ化される。各ソフトウエア・セル１０２はデータ及びアプリケーションの双方を含む、あるいは含み得る。各ソフトウエア・セルはまた、ネットワーク１０４とシステム１０１全体の中でセルをグローバルに識別するＩＤが含まれる。ソフトウエア・セルのこの構造的均一性と、ネットワークの中でのソフトウエア・セルの一意的識別によって、ネットワークの任意のコンピュータ又はコンピューティング・デバイスでのアプリケーションの処理が改善される。例えば、クライアント１０６はソフトウエア・セル１０２の作成を行うこともできるが、クライアント１０６の処理能力は限られていることから、このソフトウエア・セルをサーバー１０８へ伝送して、処理してもらうこともできる。従って、ソフトウエア・セルはネットワーク１０４全体を移動してネットワーク上での処理用リソース可用性に基づく処理を行うことが可能となる。
【００１９】
また、システム１０１のプロセッサとソフトウエア・セルが均質な構造を有することから、今日の異質なネットワークの混在という問題の多くを防ぐことができる。例えば、任意の命令セットを用いる任意のどのＩＳＡ上でも、アプリケーションの処理を許容しようとする非効率なプログラミングモデル（Ｊａｖａの仮想マシーンのような仮想マシーンなど）が回避される。従って、システム１０１は今日のネットワークよりもはるかに効果的、かつ、はるかに効率的に、広帯域処理の実現が可能になる。
【００２０】
ネットワーク１０４の全てのメンバーのための基本となる処理用モジュールはプロセッサ・エレメント（ＰＥ）である。図２にＰＥの構造を例示する。この図に示すように、ＰＥ２０１は処理ユニット（ＰＵ）２０３、ＤＭＡＣ２０５、複数の付加処理ユニット（ＡＰＵ）、すなわち、ＡＰＵ２０７、ＡＰＵ２０９、ＡＰＵ２１１、ＡＰＵ２１３、ＡＰＵ２１５、ＡＰＵ２１７、ＡＰＵ２１９、ＡＰＵ２２１、を具備する。ローカルＰＥバス２２３はＡＰＵと、ＤＭＡＣ２０５と、ＰＵ２０３との間でデータとアプリケーションを伝送する。ローカルＰＥバス２２３は従来のアーキテクチャなどを備えていてもよいし、又はパケット交換式ネットワークとして実装されてもよい。パケット交換式ネットワークとして実装される場合は、より多くのハードウエアが必要となり、その一方で利用可能な帯域幅が増加する。
【００２１】
ＰＥはデジタル論理回路を実装する様々な方法を用いて構成可能である。しかしながら、ＰＥ２０１は好適にはシリコン基板上の相補型金属酸化膜半導体（ＣＭＯＳ：complementary metal oxide semiconductor）を用いている単一の集積回路として構成される。基

板用代替材料の中には、ガリウム砒素、ガリウムアルミニウム砒素、及び多種多様のドーパントを用いるその他のいわゆるＩＩＩ－Ｂ化合物が含まれる。またＰＥ２０１は超伝導材料（高速単一磁束量子（ＲＳＦＱ：rapid single-flux-quantum）論理回路など）を用いて実装されることもできる。

【００２２】

ＰＥ２０１は高帯域メモリ接続部２２７を介して、ダイナミック・ランダム・アクセス・メモリ（ＤＲＡＭ）２２５と密接に関連している。ＤＲＡＭ２２５はＰＥ２０１用メイン・メモリとして機能する。ＤＲＡＭ２２５は好適にはダイナミック・ランダム・アクセス・メモリであることが望ましいとはいえ、ＤＲＡＭ２２５は他の手段、例えば、スタティック・ランダム・アクセス・メモリ（ＳＲＡＭ）として、磁気ランダム・アクセス・メモリ（ＭＲＡＭ）、光メモリ又はホログラフィ・メモリなどを用いてＤＲＡＭ２２５を実装することも出来る。ＤＭＡＣ２０５によってＤＲＡＭ２２５と、ＰＥ２０１のＡＰＵとＰＵとの間のデータ伝送が改善される。以下に更に説明するように、ＤＭＡＣ２０５によって、各ＡＰＵに対するＤＲＡＭ２２５内の排他的領域が指定されるが、この排他的領域の中へはＡＰＵだけしかデータの書き込みができず、また、ＡＰＵだけしかこの排他的領域からのデータ読出しを行うことができない。また、この排他的領域は“サンドボックス”と呼ばれる。

【００２３】

ＰＵ２０３は、データ及びアプリケーションのスタンド・アローン型処理が可能な、標準的プロセッサなどであってよい。作動時に、ＰＵ２０３はＡＰＵによってデータ及びアプリケーションの処理のスケジュール管理と全般的管理とを行う。ＡＰＵは好適には単一命令、複数データ（ＳＩＭＤ：single instruction, multiple data）プロセッサであることが望ましい。ＰＵ２０３の制御によって、ＡＰＵは並列的かつ独立に、これらのデータ及びアプリケーションの処理を実行する。ＤＭＡＣ２０５は、共用ＤＲＡＭ２２５に格納されているデータとアプリケーションへのＰＵ２０３とＡＰＵによるアクセス制御を行う。ＰＥ２０１は好適には８個のＡＰＵを含むことが望ましいとはいえ、必要とする処理パワーに応じて、ＰＥ内でこの数より多少上下する個数のＡＰＵを用いてもよい。ＰＵ２０３とＡＰＵの一部あるいは全ては、同じハードウエア構造、及び／又は機能を有することが可能である。個々のプロセッサは必要に応じて、ソフトウエアによって制御を行う、又は制御されるプロセッサとして構成されうる。例えば図３では、ＰＥ２０１は同じアーキテクチャを持つ９個のプロセッサを含むことができる。９個のプロセッサのうちの１つのプロセッサは、制御を行うプロセッサ（例：ＰＵ２０３）と呼ばれ、残りのプロセッサは制御されるプロセッサと呼ばれる（ＡＰＵ２０７、２０９、２１１、２１３、２１５、２１７、２１９、２１２など）。また、ＰＥ２０１のようないくつかのＰＥを結合（まとめてパッケージ化）して処理パワーの改善を図ることができる。

【００２４】

例えば、図３に示すように、一つ以上のチップ・パッケージなどの中に４つのＰＥをパッケージ化してネットワーク１０４のメンバー用の単一プロセッサを形成してもよい。この構成は広帯域エンジン（ＢＥ）と呼ばれる。図３に示すように、ＢＥ３０１には４つのＰＥ（ＰＥ３０３、ＰＥ３０５、ＰＥ３０７、及びＰＥ３０９）が含まれる。これらのＰＥ間の通信は、ＢＥバス３１１を介して行われる。広帯域メモリ接続部３１３によって共用ＤＲＡＭ３１５とこれらのＰＥ間の通信が行われる。ＢＥバス３１１の代わりに、ＢＥ３０１のＰＥ間の通信は、ＤＲＡＭ３１５とこのメモリ接続部を介して行うことができる。

【００２５】

入力／出力（Ｉ／Ｏ）インターフェース３１７と外部バス３１９とは、広帯域エンジン３０１とネットワーク１０４の他のメンバー間で通信を行う。ＢＥ３０１の各ＰＥは、ＰＥのＡＰＵによって行われるアプリケーションとデータの並列的かつ独立した処理と同様の並列的かつ独立した方法で、データとアプリケーションの処理を実行する。

【００２６】

図４にＡＰＵの構造を例示する。ＡＰＵ４０２にはローカル・メモリ４０６、レジスタ４１０、４つの浮動小数点ユニット４１２、及び４つの整数演算ユニット４１４が含まれる。しかし、ここでもまた必要とする処理パワーに応じて、４個より多少上下する個数の浮動小数点演算ユニット４１２と整数演算ユニット４１４を用いてもよい。１つの好ましい実施形態では、ローカル・メモリ４０６には１２８キロバイトの記憶容量が含まれ、レジスタ４１０の容量は１２８×１２８ビットである。浮動小数点ユニット４１２は、毎秒３２０億浮動小数点演算（３２ＧＬＰＯＳ）で好適に作動し、整数ユニット４１４は毎秒３２０億回の演算速度（３２ＧＯＰ）で好適に作動する。

【００２７】

ローカル・メモリ４０２はキャッシュ・メモリではない。ローカル・メモリ４０２は好適にはＳＲＡＭとして構成されることが望ましい。ＡＰＵに対するキャッシュ・コヒーレンシー、つまりキャッシュの整合性のサポートは不要である。ＰＵは当該ＰＵで開始されるダイレクト・メモリ・アクセス（ＤＭＡ）をサポートするために、キャッシュの整合性が要求される場合もある。しかし、ＡＰＵ又は外部デバイスからの、及び外部デバイスへのアクセスに対するキャッシュの整合性は不要である。

【００２８】

ＡＰＵ４０２には更に、ＡＰＵへ及びＡＰＵからアプリケーションとデータを伝送するためのバス４０４が含まれる。１つの好ましい実施形態では、このバスは１０２４ビットの幅を持つ。ＡＰＵ４０２には更に内部バス４０８と、４２０と、４１８とが含まれる。１つの好ましい実施形態では、バス４０８は２５６ビットの幅を持ち、ローカル・メモリ４０６とレジスタ４１０間で通信を行う。バス４２０と４１８とは、それぞれ、レジスタ４１０と浮動小数点演算ユニット４１２との間、及びレジスタ４１０と整数ユニット４１４との間で通信を行う。１つの好ましい実施形態では、レジスタ４１０から浮動小数点演算ユニット４１２又は整数ユニットへのバス４１８の幅は３１８ビットであり、浮動小数点演算ユニット４１２又は整数演算ユニット４１４からレジスタ４１０へのバス４１８と４２０の幅は１２８ビットである。浮動小数点演算ユニット４１２又は整数演算ユニット４１４への幅より広い、レジスタ４１０から浮動小数点演算ユニットまたは整数演算ユニットへの上記バスの広い幅によって、レジスタ４１０からのより広いデータ・フローが処理中に許容される。各計算には最大３ワードが必要になる。しかし、各計算結果は一般に１ワードだけである。

【００２９】

図５－１０は、ネットワーク１０４のメンバーのプロセッサのモジュラー構造を更に例示する図である。例えば、図５に示すように、１つのプロセッサには単一のＰＥ５０２を含むことができる。上述のように、このＰＥには一般に、ＰＵと、ＤＭＡＣと、８個のＡＰＵとが含まれる。各ＡＰＵにはローカル・ストレージ（ＬＳ）が含まれる。一方、プロセッサはビジュアライザ（ＶＳ）５０５の構造を有する。図５に示すように、ＶＳ５０５はＰＵ５１２と、ＤＭＡＣ５１４と、４つのＡＰＵ（ＡＰＵ５１６、ＡＰＵ１１８、ＡＰＵ５２０、ＡＰＵ５２２）とを有する。ＰＥのその他の４つのＡＰＵによって通常占領されるチップ・パッケージ内のスペースは、この場合、ビクセル・エンジン５０８、画像用キャッシュ５１０、及びブラウン管コントローラ（ＣＲＴＣ）５０４によって占領される。ＰＥ５０２又はＶＳ５０５に求められる通信速度に応じて、チップ・パッケージの中に光インターフェース５０６が含まれる場合もある。

【００３０】

この標準化されたモジュラー構造を用いて、多数の他のプロセッサが、容易にかつ効率的に構成されうる。例えば、図６に示すプロセッサは、２つのチップ・パッケージ（ＢＥを備えるチップ・パッケージ６０２と、４つのＶＳを含むチップ・パッケージ６０４）を有する。入出力部（Ｉ／Ｏ）６０６によって、チップ・パッケージ６０２のＢＥとネットワーク１０４との間にインターフェースが設けられる。バス６０８はチップ・パッケージ６０２とチップ・パッケージ６０４との間で通信を行う。入出力プロセッサ（ＩＯＰ）６１０によってデータ・フローが制御され、Ｉ／Ｏ６０６への、またはＩ／Ｏ６０６からの

入出力が行われる。Ｉ／Ｏ６０６は特定用途向集積回路（ＡＳＩＣ：application specific integrated circuit）として製造が可能である。ＶＳからの出力はビデオ信号６１２である。

【００３１】

図７に、ネットワーク１０４のその他のメンバーへ超高速通信を行う２つの光インターフェース７０４と７０６とを備えたＢＥ７０２用のチップ・パッケージ（またはローカルに接続された、その他のチップ・パッケージ）を例示する。ＢＥ７０２はネットワーク１０４のサーバーなどとして機能することができる。

【００３２】

図８のチップ・パッケージは２つのＰＥ８０２と８０４、及び２つのＶＳ８０６と８０８を有する。Ｉ／Ｏ８１０はチップ・パッケージとネットワーク１０４との間にインターフェースを与える。チップ・パッケージからの出力はビデオ信号１である。この構成は画像処理用ワークステーションなどとして機能することができる。

【００３３】

図９に更に別の構成を例示する。この構成は、図８に例示されている構成の処理パワーの１／２を含む。２つのＰＥの代わりに、１つのＰＥ９０２が設けられ、２つのＶＳの代わりに１つのＶＳ９０４が設けられる。Ｉ／Ｏ９０６は図８に例示されているＩ／Ｏの帯域幅の１／２の帯域幅を有する。しかしこのようなプロセッサは、画像処理用ワークステーションとしても機能することができる。

【００３４】

最後の構成を図１０に示す。このプロセッサは単一のＶＳ１００２とＩ／Ｏ１００４だけから構成される。この構成はＰＤＡなどとして機能することができる。

【００３５】

図１１Ａにネットワーク１０４のプロセッサのチップ・パッケージの中への光インターフェースの統合を例示する図を示す。これらの光インターフェースによって、光信号は電気信号に変換され、電気信号は光信号へ変換される。また、これらの光インターフェースは、ガリウム砒素、アルミニウムガリウム砒素、ゲルマニウム、及びその他の元素や化合物などを含む様々な材料から構成されうる。この図に示すように、光インターフェース１１０４と１１０６はＢＥ１１０２のチップ・パッケージ上に組み立てられる。ＢＥバス１１０８はＢＥ１１０２のＰＥ、すなわち、ＰＥ１１１０、ＰＥ１１１２、ＰＥ１１１４、ＰＥ１１１６及びこれらの光インターフェースとの間で通信を行う。光インターフェース１１０４には２つのポート（ポート１１１８とポート１１２０）が含まれ、また光インターフェース１１０６には２つのポート（ポート１１２２とポート１１２４）が含まれる。ポート１１１８、１１２０、１１２２及び１１２４は光導波路１１２６、１１２８、１１３０、１１３２とそれぞれ接続される。光信号は光インターフェース１１０４と１１０６のポートを介して、これらの光導波路の中を通り、ＢＥ１１０２へ、及びＢＥ１１０２から伝送される。

【００３６】

このような光導波路と各ＢＥの４つの光ポートとを用いて、様々な構成において複数のＢＥをまとめて接続してもよい。例えば図１１Ｂに示すように、このような光ポートを介して２つまたはそれ以上のＢＥ（ＢＥ１１５２、ＢＥ１１５４、ＢＥ１１５６など）を直列に接続することができる。この例では、ＢＥ１１５２の光インターフェース１１６６は、その光ポートを介してＢＥ１１５４の光インターフェース１１６０の光ポートと接続される。同様に、ＢＥ１１５４の光インターフェース１１６２の光ポートは、ＢＥ１１５６の光インターフェース１１６４の光ポートと接続される。

【００３７】

図１１Ｃにマトリクス構成を例示する。この構成では、各ＢＥの光インターフェースは２つの他のＢＥと接続される。この図に示すように、ＢＥ１１７２の光インターフェース１１８８の光ポートの中の１つが、ＢＥ１１７６の光インターフェース１１８２の光ポートと接続される。光インターフェース１１８８のもう一方の光ポートは、ＢＥ１１７８の

光インターフェース１１８４光ポートと接続される。同様に、ＢＥ１１７４の光インターフェース１１９０の１つの光ポートは、ＢＥ１１７８の光インターフェース１１８４のもう一方の光ポートと接続される。光インターフェース１１９０のもう一方の光ポートは、ＢＥ１１８０の光インターフェース１１８６の光ポートと接続される。このマトリクス構成は他のＢＥに対しても同様に拡張することができる。

【００３８】

シリアル構成かマトリクス構成のいずれかを用いて、任意の所望のサイズとパワーからなるネットワーク１０４用プロセッサ構成が可能になる。言うまでもなく、ＢＥの光インターフェースに対して、あるいはＢＥよりＰＥ数が上下するプロセッサに対して追加ポートを加えて他の構成を形成してもよい。

【００３９】

図１２ＡにＢＥのＤＲＡＭに対する制御システムと構造を例示する。同様の制御システムと構造が、別のサイズを持ち、多少異なる数のＰＥを含むプロセッサの中で用いられる。この図に示すように、クロスバ交換機によって、ＢＥ１２０１を備える４つのＰＥからなるＤＭＡＣ１２１０が８つのバンク・コントロール１２０６と接続される。各バンク・コントロール１２０６によってＤＲＡＭ１２０４の８つのバンク１２０８（４つだけしか図示されていない）が制御される。したがって、ＤＲＡＭ１２０４は合計で６４のバンクを具備することになる。好ましい実施形態では、ＤＲＡＭ１２０４は６４メガバイトの容量を持ち、各バンクは１メガバイトの容量を持つ。各バンク内の最小のアドレス指定可能単位は、この好ましい実施形態では１０２４ビットのブロックである。

【００４０】

ＢＥ１２０１にはスイッチ・ユニット１２１２も含まれる。スイッチ・ユニット１２１２により、ＢＥ１２０１に密接に接続されているＢＥの他のＡＰＵの、ＤＲＡＭ１２０４へのアクセスが可能となる。したがって、第２のＢＥを第１のＢＥと密接に接続することが可能となり、さらに、各ＢＥの各ＡＰＵは、ＡＰＵが通常アクセス可能なメモリ・ロケーション数の２倍のアドレス指定を行うことが可能となる。スイッチ・ユニット１２１２のようなスイッチ・ユニットを介して、第１のＢＥのＤＲＡＭから第２のＢＥのＤＲＡＭへのデータの直接読み出し、または、第２のＢＥのＤＲＡＭから第１のＢＥのＤＲＡＭへのデータの直接書き込みを行うことが可能となる。

【００４１】

例えば、図１２Ｂに示すように、このような書き込みを行うために、第１のＢＥのＡＰＵ（ＢＥ１２２２のＡＰＵ１２２０など）によって、第２のＢＥのＤＲＡＭ（通常の場合のようなＢＥ１２２２のＢＥ１２２２のＤＲＡＭ１２２４ではなく、ＢＥ１２２６のＤＲＡＭ１２２８など）のメモリ・ロケーションへの書き込みコマンドが出される。ＢＥ１２２２のＤＭＡＣ１２３０は、クロスバ交換機１２２１を介してバンク・コントロール１２３４へ書き込みコマンドを送り、バンク・コントロール１２３４はバンク・コントロール１２３４と接続された外部ポート１２３２へコマンドを伝送する。ＢＥ１２２６のＤＭＡＣ１２３８は書き込みコマンドを受け取り、このコマンドをＢＥ１２２６のスイッチ・ユニット１２４０へ転送する。スイッチ・ユニット１２４０は書き込みコマンドの中に含まれるＤＲＡＭアドレスを識別し、ＢＥ１２２６のバンク・コントロール１２４２を介して、ＤＲＡＭ１２２８のバンク１２４４へ、ＤＲＡＭアドレス内に格納するデータを送る。したがって、スイッチ・ユニット１２４０により、ＤＲＡＭ１２２４とＤＲＡＭ１２２８の双方は、ＢＥ１２２２のＡＰＵ用の単一メモリ空間として機能することが可能になる。

【００４２】

図１３にＤＲＡＭの６４個のバンク構成を図示する。これらのバンクは８つの行（１３０２、１３０４、１３０６、１３０８、１３１０、１３１２、１３１４、１３１６）と８つの列（１３２０、１３２２、１３２４、１３２６、１３２８、１３３０、１３３２、１３３４）とで構成されている。各行はバンク・コントローラにより制御される。したがって、各バンク・コントローラは８メガバイトのメモリを制御する。

【００４３】

図１４Ａと１４Ｂに最小のアドレス指定可能な格納単位（１０２４ビットのブロックなど）でのＤＲＡＭの格納とアクセスを行うための異なる構成を例示する。図１４Ａでは、ＤＭＡＣ１４０２は単一のバンク１４０４の中に８つの１０２４ビット・ブロック１４０６を格納する。一方で図１４Ｂでは、ＤＭＡＣ１４１２によって、１０２４ビットを含むデータ・ブロックの読み出しと書き込みが行われるが、これらのブロックは２つのバンク（バンク１４１４とバンク１４１６）の間で分配される。したがって、これらのバンクの各々には１６個のデータ・ブロックが含まれ、データの各ブロックには５１２ビットが含まれる。この分配によって、ＤＲＡＭのアクセスをさらに高速なものに改善することが可能となり、ある種のアプリケーションの処理に役立つ。

【００４４】

図１５にＰＥ内のＤＭＡＣ１５０４のアーキテクチャを例示する。この図に示されているように、各ＡＰＵがＤＭＡＣ１５０６の構造上のノード１５０４へ直接アクセスを行うように、ＤＭＡＣ１５０６を含む構造上のハードウエアはＰＥを通じて配設される。各ノードは、ノードが直接アクセスを行う対象のＡＰＵによるメモリ・アクセスに適した論理処理を実行する。

【００４５】

図１６にＤＭＡＣの他の実施形態、すなわち、非分散型アーキテクチャを例示する。この場合、ＤＭＡＣ１６０６の構造上のハードウエアは集中型である。ＡＰＵ１６０２とＰＵ１６０４は、ローカルＰＥバス１６０７を介してＤＭＡＣ１６０６を用いて通信を行う。ＤＭＡＣ１６０６はクロスバ交換機を介して、バス１６０８と接続される。バス１６０８はＤＲＡＭ１６１０と接続されている。

【００４６】

上述のように１つのＰＥの複数のＡＰＵの全ては、独立に、共用ＤＲＡＭ内のデータへのアクセスが可能である。その結果、第１のＡＰＵがあるデータをそのローカル・ストレージで処理しているときに、第２のＡＰＵがこれらのデータを要求する場合もある。その時点で共用ＤＲＡＭから第２のＡＰＵへデータが出力された場合、データの値を変化させうる第１のＡＰＵの進行中の処理に起因して、そのデータが無効になる場合がある。したがって、その時点で第２のプロセッサが共用ＤＲＡＭからデータを受け取った場合、第２のプロセッサでエラー結果が生じる恐れがある。例えば、このようなデータとしては、グローバル変数用の具体的な値が上げられる。第１のプロセッサがその処理中にその値を変えた場合、第２のプロセッサはもう使用されていない値を受け取ることになる。したがって、共用ＤＲＡＭの範囲内で、メモリ・ロケーションからの、及び、メモリ・ロケーションへのＡＰＵによるデータの読み出しと書き込みを同期させる何らかの方式が必要となる。この方式では、別のＡＰＵがそのローカル・ストレージで現在働きかけている対象データであって、したがって最新のものではないデータのメモリ・ロケーションからの読み出しと、最新のデータを格納するメモリ・ロケーションの中へのデータの書き込みと、を行わないようにする必要がある。

【００４７】

これらの問題を解決するために、ＤＲＡＭの各アドレス指定が可能なメモリ・ロケーションに対して、そのメモリ・ロケーションに格納されているデータに関連する状態情報を格納するために、ＤＲＡＭの中でメモリの追加セグメントの割り振りが行われる。この状態情報の中には、フル・エンプティ（Ｆ／Ｅ）ビットと、メモリ・ロケーションからデータを要求するＡＰＵの識別子（ＡＰＵ　ＩＤ）と、要求されたデータを読み出す読み出し先となるＡＰＵのローカル・ストレージのアドレス（ＬＳアドレス）とが含まれる。ＤＲＡＭのアドレス指定可能なメモリ・ロケーションは任意のサイズとすることができる。ある好ましい実施形態では、このサイズは１０２４ビットである。

【００４８】

Ｆ／Ｅビットの１への設定は、関連するメモリ・ロケーションに格納されているデータが最新のものであることを示す。一方、Ｆ／Ｅビットの０への設定は、関連するメモリ・ロケーションに格納されているデータが最新のものではないことを示す。このビットが０

に設定されているときにＡＰＵがデータを要求しても、ＡＰＵによってそのデータの即時読み出しは妨げられる。この場合、そのデータを要求しているＡＰＵを識別するＡＰＵＩＤと、データが最新のものになっているとき、そのデータを読み出す読み出し先となるこのＡＰＵローカル・ストレージ内のメモリ・ロケーションを識別するＬＳアドレスとが、追加メモリ・セグメントの中へ入力される。

【００４９】

また追加メモリ・セグメントは、ＡＰＵのローカル・ストレージ内の各メモリ・ロケーションに対して割り振られる。この追加メモリ・セグメントは“ビジー・ビット”と呼ばれる１ビットを格納する。ビジー・ビットはＤＲＡＭから検索される固有データの格納用として、関連するＬＳメモリ・ロケーションの予約を行うために使用される。ローカル・ストレージの特定のメモリ・ロケーションに対してビジー・ビットが１に設定されている場合、これらの固有データの書き込み用としてのみ、ＡＰＵはこのメモリ・ロケーションを使用することができる。一方、ローカル・ストレージの特定のメモリ・ロケーションに対して、ビジー・ビットが０に設定されている場合、ＡＰＵは任意のデータの書き込み用にこのメモリ・ロケーションを使用することができる。

【００５０】

Ｆ／Ｅビット、ＡＰＵ　ＩＤ、ＬＳアドレス、及びビジー・ビットが、ＰＥの共用ＤＲＡＭからの、及び、ＰＥの共用ＤＲＡＭへのデータの読み出しと書き込みを同期させるために使用される方法の例を、図１７Ａから図１７Ｏに例示する。

【００５１】

図１７Ａに示すように、１つ以上のＰＥ（ＰＥ１７２０など）がＤＲＡＭ１７０２を使用する。ＰＥ１７２０にはＡＰＵ１７２２とＡＰＵ１７４０とが含まれる。ＡＰＵ１７２２には制御論理回路１７２４が含まれ、ＡＰＵ１７４０には制御論理回路１７４２が含まれる。ＡＰＵ１７２２にはローカル・ストレージ１７２６も含まれる。このローカル・ストレージには複数のアドレス可能なメモリ・ロケーション１７２８が含まれる。ＡＰＵ１７４０にはローカル・ストレージ１７４４が含まれ、このローカル・ストレージにも複数のアドレス可能なメモリ・ロケーション１７４６が含まれる。これらのアドレス可能なメモリ・ロケーションのすべては、好適にはサイズが１０２４ビットであることが望ましい。

【００５２】

メモリの追加セグメントは各ＬＳのアドレス可能なメモリ・ロケーションと関連付けられる。例えば、メモリ・セグメント１７２９と１７３４とはそれぞれ、ローカルなメモリ・ロケーション１７３１と１７３２と関連付けられ、メモリ・セグメント１７５２はローカル・メモリ・ロケーション１７５０と関連付けられる。上述のような“ビジー・ビット”はこれらの追加のメモリ・セグメントの各々の中に格納される。ローカル・メモリ・ロケーション１７３２は、このメモリ・ロケーションがデータを含むことを示すいくつかのＸ印を用いて示されている。

【００５３】

ＤＲＡＭ１７０２には、メモリ・ロケーション１７０６と１７０８とを含む、複数のアドレス可能なメモリ・ロケーション１７０４を含まれる。またこれらのメモリ・ロケーションは、好適にはサイズが１０２４ビットであることが望ましい。メモリの追加セグメントはまた、これらのメモリ・ロケーションの各々とも関連付けられる。例えば、追加のメモリ・セグメント１７６０はメモリ・ロケーション１７０６と関連付けられ、また、追加のメモリ・セグメント１７６２はメモリ・ロケーション１７０８と関連付けられる。各メモリ・ロケーションに格納されるデータに関連する状態情報は、メモリ・ロケーションに関連付けられたメモリに格納される。この状態情報の中には、上述のように、Ｆ／Ｅビット、ＡＰＵ　ＩＤ、及びＬＳアドレスが含まれる。例えば、メモリ・ロケーション１７０８については、この状態情報には、Ｆ／Ｅビット１７１２、ＡＰＵ　ＩＤ１７１４、及びＬＳアドレス１７１６が含まれる。

【００５４】

この状態情報とビジー・ビットとを用いて、ＰＥのＡＰＵ、又は１グループのＰＥ間での、共用ＤＲＡＭからの、および、同期した共用ＤＲＡＭからの読み出しと、同期した共用ＤＲＡＭへのデータの書き込みを行うことができる。

【００５５】

図１７Ｂに、ＡＰＵ１７２２のＬＳメモリ・ロケーション１７３２から、ＤＲＡＭ１７０２のメモリ・ロケーション１７０８へのデータの同期書き込みの開始を例示する図を示す。ＡＰＵ１７２２の制御論理回路１７２４によって、これらのデータの同期書き込みが開始される。メモリ・ロケーション１７０８はエンプティであるため、Ｆ／Ｅビット１７１２は０に設定される。その結果、メモリ・ロケーション１７０８の中へＬＳロケーション１７３２内のデータを書き込むことが可能となる。このビットが１に設定され、メモリ・ロケーション１７０８がフル状態であり、最新の有効データを含むことが示されている場合、制御回路１７２４はエラー・メッセージを受け取ることになり、このメモリ・ロケーションへのデータの書き込みは禁止される。

【００５６】

メモリ・ロケーション１７０８への成功したデータの同期書き込みの結果を図１７Ｃに示す。この書き込まれたデータはメモリ・ロケーション１７０８に格納され、Ｆ／Ｅビット１７１２は１に設定される。この設定により、メモリ・ロケーション１７０８がフル状態であること、及び、このメモリ・ロケーションの中のデータが最新であり有効であることが示される。

【００５７】

図１７Ｄに、ＤＲＡＭ１７０２のメモリ・ロケーション１７０８から、ローカル・ストレージ１７４４のＬＳメモリ・ロケーション１７５０へのデータの同期読み出しの開始を例示する図を示す。この読み出しを開始するために、ＬＳメモリ・ロケーション１７５０のメモリ・セグメント１７５２の中のビジー・ビットが１に設定されて、このデータに対するメモリ・ロケーションが予約される。このビジー・ビットを１に設定することにより、ＡＰＵ１７４０がこのメモリ・ロケーションにその他のデータを格納することはなくなっている。

【００５８】

図１７Ｅに示すように、制御論理回路１７４２は次に、ＤＲＡＭ１７０２のメモリ・ロケーション１７０８に対し、同期読取りコマンドを出す。このメモリ・ロケーションに関連するＦ／Ｅビット１７１２は１に設定されているので、メモリ・ロケーション１７０８に格納されるデータは最新の有効データであると見なされる。その結果、メモリ・ロケーション１７０８からＬＳメモリ・ロケーション１７５０へのデータ転送の準備の際に、Ｆ／Ｅビット１７１２は０に設定される。この設定は図１７Ｆに示されている。このビットを０に設定することは、これらのデータの読み出しの後に、メモリ・ロケーション１７０８のデータは無効になることを示す。

【００５９】

図１７Ｇに示すように、メモリ・ロケーション１７０８内のデータは次に、メモリ・ロケーション１７０８からＬＳメモリ・ロケーション１７５０へ読み出される。図１７Ｈは最終状態を示す。メモリ・ロケーション１７０８のデータのコピーはＬＳメモリ・ロケーションに格納される。Ｆ／Ｅビット１７１２は０に設定され、メモリ・ロケーション１７０８のデータは無効であることが示される。この無効はＡＰＵ１７４０によって行われた上記データの変更の結果である。メモリ・セグメント１７５２のビジー・ビットも０に設定される。この設定によって、ＡＰＵ１７４０がＬＳメモリ・ロケーション１７５０を任意の目的に利用できること、すなわち、このＬＳメモリ・ロケーションがもはや固有データの受信を待機している予約状態ではないことが示される。したがって、ＬＳメモリ・ロケーション１７５０は任意の目的のために、ＡＰＵ１７４０によるＬＳメモリ・ロケーション１７５０へのアクセスが可能になる。

【００６０】

図１７Ｉから１７Ｏに、ＤＲＡＭ１７０２のメモリ・ロケーション用Ｆ／Ｅビットが０

に設定され、このメモリ・ロケーションのデータが最新のものでもなく有効なものでもないことが示されている場合の、ＤＲＡＭ１７０２（メモリ・ロケーション１７０８など）のメモリ・ロケーションから、ＡＰＵのローカル・ストレージ（ローカル・ストレージ１７４４のＬＳメモリ・ロケーション１７５２など）のＬＳメモリ・ロケーションへのデータの同期読み出しが例示されている。図１７Ｉに示すように、この転送を開始するために、ＬＳメモリ・ロケーション１７５０のメモリ・セグメント１７５２内のビジー・ビットは１に設定され、このデータ転送用としてこのＬＳメモリ・ロケーションが予約される。図１７Ｊに示すように、制御論理回路１７４２は次に、ＤＲＡＭ１７０２のメモリ・ロケーション１７０８に対し、同期読取りコマンドを出す。このメモリ・ロケーションと関連付けられたＦ／Ｅビット（Ｆ／Ｅビット１７１２）は０に設定されているので、メモリ・ロケーション１７０８に格納されているデータは無効である。その結果、信号は制御論理回路１７４２へ伝送され、このメモリ・ロケーションからのデータの即時読み出しが阻止される。

【００６１】

図１７Ｋに示すように、ＡＰＵ　ＩＤ１７１４とこの読取コマンド用のＬＳアドレス１７１６は次にメモリ・セグメントの中へ書き込まれる。この場合、ＡＰＵ１７４０用のＡＰＵ　ＩＤとＬＳメモリ・ロケーション１７５０用のＬＳメモリロケーションは、メモリ・セグメント１７６２の中へ書き込まれる。したがって、メモリ・ロケーション１７０８内のデータが最新のものになっているとき、このＡＰＵ　ＩＤとＬＳメモリ・ロケーションは、最新のデータを伝送する伝送先のメモリ・ロケーションを決定するために使用される。

【００６２】

メモリ・ロケーション１７０８内のデータは、ＡＰＵがこのメモリ・ロケーションの中へデータを書き込むと、有効で最新のデータとなる。ＡＰＵ１７２２のメモリ・ロケーション１７３２などから、メモリ・ロケーション１７０８への、データの同期書き込みが図１７Ｌに示されている。このメモリ・ロケーション用のＦ／Ｅビット１７１２が０に設定されているため、これらのデータのこの同期書き込みは許される。

【００６３】

図１７Ｍに示すように、この書き込み後、メモリ・ロケーション１７０８のデータは最新の有効データになる。したがって、メモリ・セグメント１７６２から得られるＡＰＵ　ＩＤ１７１４とＬＳアドレス１７１６とは、メモリ・セグメント１７６２から即座に読み出され、次いで、この情報はこのセグメントから削除される。メモリ・ロケーション１７０８のデータの即時読み出しを予測して、Ｆ／Ｅビット１７１２もまた０に設定される。図１７Ｎに示すように、ＡＰＵ　ＩＤ１７１４とＬＳアドレス１７１６とを読み出すと、ＡＰＵ１７４０のＬＳメモリ・ロケーション１７５０へメモリ・ロケーション１７０８内の有効データを読み出すために、この情報は直ちに使用される。図１７０に最終状態を示す。この図は、メモリ・ロケーション１７０８からメモリ・ロケーション１７５０にコピーされた有効データと、０に設定されたメモリ・セグメント１７５２内のビジー・ビットと、０に設定されたメモリ・セグメント１７６２内のＦ／Ｅビット１７１２とを示す。このビジー・ビットを０に設定することにより、任意の目的のためにＡＰＵ１７４０がＬＳメモリ・ロケーション１７５０のアクセスを行うことができる。このＦ／Ｅビットを０に設定することにより、メモリ・ロケーション１７０８内のデータがもはや最新のものではなく、有効なものでもないことが示される。

【００６４】

図１８は上述のオペレーションと、ＤＲＡＭのメモリ・ロケーションの様々な状態とを要約する図であり、この状態はＦ／Ｅビットの状態と、ＡＰＵ　ＩＤと、メモリ・ロケーションに対応するメモリ・セグメントの中に格納されたＬＳアドレスとに基づく。このメモリ・ロケーションは、３つの状態を持つことが可能である。これらの３つの状態として、Ｆ／Ｅビットが０に設定され、ＡＰＵ　ＩＤまたはＬＳアドレスに対して情報が提供されないエンプティ状態１８８０と、Ｆ／Ｅビットが１に設定され、ＡＰＵ　ＩＤまたはＬ

Ｓアドレスに対して情報が提供されないフル状態１８８２と、Ｆ／Ｅビットが０に設定され、ＡＰＵ　ＩＤとＬＳアドレスに対して情報が提供されるブロッキング状態１８８４とがある。

【００６５】

この図に示すように、エンプティ状態１８８０では、同期書き込みオペレーションが許され、フル状態１８８２への遷移という結果が得られる。しかし、メモリ・ロケーションがエンプティ状態であるときは、メモリ・ロケーション内のデータが最新のものではないので、同期読み出しオペレーションに対しては、ブロッキング状態へ遷移するという結果となる。

【００６６】

フル状態１８８２では、同期読み出しオペレーションが許され、エンプティ状態１８８０への遷移という結果が得られる。一方、有効データの上書きを避けるために、フル状態１８８２の同期書き込みオペレーションは禁止される。このような書き込みオペレーションがこの状態で試みられる場合、状態の変化は生じず、エラー・メッセージがＡＰＵの対応する制御論理回路へ伝送される。

【００６７】

ブロッキング状態１８８４では、メモリ・ロケーションの中へのデータの同期書き込みが許され、エンプティ状態１８８０への遷移という結果が得られる。一方、ブロッキング状態１８８４での同期読み出しオペレーションは禁止される。それは、このブロッキング状態を生じさせることとなった前回の同期読み出しオペレーションとのコンフリクトを阻止するためである。同期読み出しオペレーションがブロッキング状態１８８４で試みられた場合、状態変化は生じず、ＡＰＵの対応する論理制御回路へエラー・メッセージが伝送される。

【００６８】

共用ＤＲＡＭからのデータの同期読み出しと、共用ＤＲＡＭへのデータの同期書き込みを行う上述の方式は、外部デバイスからのデータ読み出しと、外部デバイスのデータ書き込み用プロセッサとして通常専用の計算用リソースを省くためにも利用することができる。この入力／出力（Ｉ／Ｏ）機能はＰＵによって行うこともできる。しかし、この同期方式の変更を利用して、適切なプログラムを実行するＡＰＵがこの機能を実行してもよい。例えば、この方式を利用して、外部デバイスによって開始されたＩ／Ｏインターフェースからのデータ伝送を求める割込み要求を受け取るＰＵは、このＡＰＵにこの要求処理を委任してもよい。次にＡＰＵは、Ｉ／Ｏインターフェースに対して同期書き込みコマンドを出す。今度はこのインターフェースによって、現在ＤＲＡＭの中へデータを書き込むことができる旨の信号が外部デバイスへ送られる。次にＡＰＵはＤＲＡＭへ同期読取りコマンドを出し、ＤＲＡＭの関連するメモリ空間をブロッキング状態に設定する。ＡＰＵはまた、データを受け取る必要があるＡＰＵのローカル・ストレージのメモリ・ロケーションに対して、ビジー・ビットをＩに設定する。ブロッキング状態では、ＤＲＡＭの関連するメモリ空間と関連付けられた追加メモリ・セグメントの中に、ＡＰＵのＩＤとＡＰＵのローカル・ストレージの関連するメモリ・ロケーションのアドレスが含まれる。次に外部デバイスは同期書き込みコマンドを出し、ＤＲＡＭの関連するメモリ空間へデータが直接書き込まれる。このメモリ空間はブロッキング状態にあるので、データは、この空間の中から、追加メモリ・セグメントの中で識別されたＡＰＵのローカル・ストレージのメモリ・ロケーションの中へ直ちに読み出される。次いで、これらのメモリ・ロケーション用のビジー・ビットは０に設定される。外部デバイスがデータの書き込みを完了したとき、ＡＰＵは伝送が完了した旨を示す信号をＰＵへ出す。

【００６９】

したがって、この方式を用いて、ＰＵに対する最小の計算上の負荷で、外部デバイスからのデータ転送処理を行うことができる。しかし、この機能を任されたＡＰＵはＰＵに対して割込み要求を出せることが望ましく、外部デバイスがＤＲＡＭに対して直接アクセスを行うことが望ましい。

【００７０】
各ＰＥのＤＲＡＭには複数の”サンドボックス”が含まれる。サンドボックスによって共用ＤＲＡＭ領域が画定され、この領域を超えて、特定のＡＰＵまたは１組のＡＰＵがデータの読み出しや書き込みを行うことはできない。これらのサンドボックスによって、１つのＡＰＵが処理するデータに起因する、別のＡＰＵによって処理されるデータの破損に対するセキュリティが与えられる。またこれらのサンドボックスによって、ソフトウエア・セルが全ＤＲＡＭの中でデータの破損を生じる可能性はなく、ネットワーク１０４から特定のサンドボックスの中へソフトウエア・セルのダウンロードを行うことが許される。本発明では、サンドボックスはＤＲＡＭとＤＭＡＣとからなるハードウエアの中に設けられる。ソフトウエアの代わりに、このハードウエア内にこれらのサンドボックスを設けることにより、速度とセキュリティという利点が得られる。

【００７１】
ＰＥのＰＵはＡＰＵへ割り当てられるサンドボックスの制御を行う。ＰＵは通常、オペレーティング・システムのような信頼のおけるプログラムだけしか作動させないので、この方式によって、セキュリティが危険にさらされることはない。この方式に従って、ＰＵはキー管理テーブルの構築と維持とを行う。図１９にこのキー管理テーブルを例示する。この図に示すように、キー管理テーブル１９０２の各エントリには、ＡＰＵ用の識別子（ＩＤ）１９０４と、そのＡＰＵ用のＡＰＵキー１９０６と、キー・マスク１９０８とが含まれる。このキー・マスクの用途について以下に説明する。キー管理テーブル１９０２は好適にはスタティック・ランダム・アクセス・メモリ（ＳＲＡＭ）のような比較的高速のメモリに格納され、ＤＭＡＣと関連付けられる。キー管理テーブル１９０２へのエントリはＰＵによって制御される。ＡＰＵがＤＲＡＭの特定格納位置（ストレージロケーション）へのデータの書き込みとあるいはＤＲＡＭの特定の格納位置からのデータの読み出しを要求すると、ＤＭＡＣはその格納位置と関連付けられたメモリ・アクセス・キーに対してキー管理テーブル１９０２内のそのＡＰＵへ割り当てられたＡＰＵキー１９０６の評価を行う。

【００７２】
図２０に示すように、ＤＲＡＭ２００２の各アドレス可能な格納位置２００６に対して専用メモリ・セグメント２０１０が割り当てられる。この格納位置用のメモリ・アクセス・キー２０１２はこの専用メモリ・セグメントの中に格納される。上述のように、やはり各アドレス可能格納位置２００６と関連付けられたさらなる追加専用メモリ・セグメント２００８によって、格納位置へのデータを書き込みと、格納位置からのデータ読み出しを行うための同期情報が格納される。

【００７３】
作動時に、ＡＰＵはＤＭＡＣへＤＭＡコマンドを出す。このコマンドにはＤＲＡＭ２００２の格納位置２００６のアドレスが含まれる。このコマンドを実行する前に、ＤＭＡＣは、キー管理テーブル１９０２におけるＡＰＵのＩＤ１９０４を用いて要求を行っているＡＰＵのキー１９０６を調べる。次いでＤＭＡＣはＡＰＵがアクセスを求める対象先であるＤＲＡＭの格納位置と関連付けられた専用メモリ・セグメント２０１０の中に格納されるメモリ・アクセス・キー２０１２と、要求を行っているＡＰＵのＡＰＵキー１９０６との比較を行う。２つのキーが一致しない場合、ＤＭＡコマンドは実行されない。一方、２つのキーが一致する場合、ＤＭＡコマンドは進行し、要求されたメモリ・アクセスが実行される。

【００７４】
図２１に他の実施形態を例示する。この実施形態では、ＰＵはメモリ・アクセス管理テーブル２１０２の維持も行う。メモリ・アクセス管理テーブル２１０２にはＤＲＡＭ内にある各サンドボックス用のエントリが含まれる。図２１の特定の例では、ＤＲＡＭには６４個のサンドボックスが含まれる。メモリ・アクセス管理テーブル２１０２内の各エントリには、サンドボックス用識別子（ＩＤ）２１０４と、ベース・メモリ・アドレス２１０６と、サンドボックス・サイズ２１０８と、メモリ・アクセス・キー２１１０と、アクセ

ス・キー・マスク２１１２とが含まれる。ベース・メモリ・アドレス２１０６によって、ＤＲＡＭにアドレスが設けられこのアドレスによって特定のメモリ・サンドボックスの最初の部分が示される。サンドボックス・サイズ２１０８によりサンドボックスのサイズが与えられ、したがって、このサイズにより特定のサンドボックスのエンドポイントが与えられる。

【００７５】

図２２はキー管理テーブル１９０２とメモリ・アクセス管理テーブル２１０２とを用いてＤＭＡコマンドを実行するためのステップを示すフローチャートである。ステップ２２０２では、ＡＰＵによってサンドボックス内の特定１つあるいは複数のメモリ・ロケーションに対するアクセス用ＤＭＡコマンドがＤＭＡＣへ出される。このコマンドにはアクセス要求を行う対象先である特定のサンドボックスの識別を行うサンドボックスＩＤ２１０４が含まれる。ステップ２２０４で、ＤＭＡＣはＡＰＵのＩＤ１９０４を利用して、キー管理テーブル１９０２内の要求を行っているＡＰＵのキー１９０６を調べる。ステップ２２０６で、ＤＭＡＣはメモリ・アクセス管理テーブル２１０２で、サンドボックスと関連付けられたメモリ・アクセス・キー２１１０を調べるコマンドで、サンドボックスＩＤ２１０４を利用する。ステップ２２０８で、ＤＭＡＣは、要求を行っているＡＰＵへ割り当てられているＡＰＵキー１９０６と、サンドボックスと関連付けられたアクセス・キー２１１０と比較する。ステップ２１１０で、２つのキーが一致するかどうかの決定が行われる。２つのキーが一致しない場合、処理はステップ２２１２へ移行し、そこでＤＭＡコマンドは先へ進まず、要求を行っているＡＰＵとＰＵのいずれか、または双方へエラーメッセージが送信される。一方、ステップ２２１０で、２つのキーの一致が得られた場合、処理はステップ２２１４へと進み、そこでＤＭＡＣはＤＭＡコマンドを実行する。

【００７６】

ＡＰＵ用キーとメモリ・アクセス・キー用のキー・マスクにより、このシステムに大きな柔軟性が与えられる。キー用のキー・マスクにより、マスクされたビットはワイルド・カードに変換される。例えば、ＡＰＵキー１９０６と関連付けられたキー・マスク１９０８が、キー・マスク１９０８内のこれらのビットを１に設定することなどにより、その最後の２ビットが“マスク”に設定されている場合、ＡＰＵキーは１又は０のいずれかになることができ、そのままメモリ・アクセス・キーに一致することになる。例えば、ＡＰＵキーは１０１０であるとする。通常、このＡＰＵキーによって１０１０のアクセス・キーを持つサンドボックスへのアクセスだけが可能になる。しかし、このＡＰＵキー用のＡＰＵキー・マスクが０００１に設定されている場合、このＡＰＵキーを用いて１０１０または１０１１のいずれかのアクセス・キーを持つサンドボックスへのアクセスを行うことが可能となる。同様に、１０１０または１０１１のいずれかのＡＰＵキーを持つＡＰＵによって、０００１に設定されたマスクを持つアクセス・キー１０１０のアクセスを行うことが可能である。ＡＰＵキー・マスクとメモリ・キーマスクの双方を同時に使用することができるので、多数のバリエーションのサンドボックスに対するＡＰＵによるアクセシビリティの設定が可能となる。

【００７７】

本発明はまた、システム１０１のプロセッサ用の新しいプログラミング・モデルも提供する。このプログラミング・モデルではソフトウエア・セル１０２が用いられる。ネットワーク１０４上の任意のプロセッサへ処理用として、これらのセルの伝送を行うことが可能である。また、この新しいプログラミング・モデルでは、システム１０１のユニークなモジュラー形アーキテクチャとシステム１０１のプロセッサとが利用される。

【００７８】

ソフトウエア・セルはＡＰＵのローカル・ストレージからＡＰＵによって直接処理される。ＡＰＵはＤＲＡＭ内のいずれのデータ又はプログラムに対しても直接働きかけることは行わない。ＤＲＡＭ内のデータとプログラムは、ＡＰＵがこれらのデータとプログラムの処理を行う前に、ＡＰＵのローカル・ストレージの中に読み込まれる。したがって、ＡＰＵのローカル・ストレージには、プログラム・カウンタと、スタックと、これらのプロ

グラムを実行するための他のソフトウエア・エレメントとが含まれる。ＰＵはＤＭＡＣに対してＤＭＡコマンドを出すことによりＡＰＵの制御を行う。

【００７９】

図２３にソフトウエア・セル１０２の構造を例示する。この図に示すように、ソフトウエア・セル２３０２などのソフトウエア・セルの中には、ルート選定情報セクション２３０４と本体部分２３０６とが含まれる。ルート選定情報セクション２３０４に含まれる情報は、ネットワーク１０４のプロトコルに依って決められる。ルート選定情報セクション２３０４にはヘッダ２３０８、宛先ＩＤ２３１０、ソースＩＤ２３１２、及び応答ＩＤ２３１４が含まれる。宛先ＩＤ２３１０にはネットワーク・アドレスが含まれる。ＴＣＰ／ＩＰプロトコルの下で、例えば、ネットワーク・アドレスはインターネット・プロトコル（Ｉ／Ｐ）である。更に、宛先ＩＤ２３１０には、処理のためにセルを伝送すべき伝送先のＰＥ及びＡＰＵの識別子が含まれる。ソースＩＤ２３１４にはネットワーク・アドレスが含まれ、このソースＩＤによってＰＥとＡＰＵとが識別され、このＰＥとＡＰＵとからセルが起動し、必要な場合に、宛先ＰＥとＡＰＵとがセルに関する追加情報を得ることが可能となる。応答ＩＤ２３１４にはネットワーク・アドレスが含まれ、この応答ＩＤ２３１４によって、セルに関するクエリとセルの処理結果とを送る送り先のＰＥとＡＰＵとが識別される。

【００８０】

セルの本体部分２３０６にはネットワークのプロトコルとは無関係の情報が含まれる。図２３の分解部分は、セルの本体部分２３０６の細部を示す。セルの本体部分２３０６のヘッダ２３２０によってセル本体部分の開始部が識別される。セル・インターフェース２３２２にはセルの利用に必要な情報が含まれる。この情報にはグローバルな一意的ＩＤ２３２４と要求されるＡＰＵ２３２６と、サンドボックス・サイズ２３２８と、前回のセルのＩＤ２３３０が含まれる。

【００８１】

グローバルな一意的ＩＤ２３２４はネットワーク１０４全体を通じてソフトウエア・セル２３０２を一意的に識別する。グローバルな一意的ＩＤ２３２４はソースＩＤ２３１２（ソースＩＤ２３１２内のＰＥ又はＡＰＵの一意的識別子など）と、ソフトウエア・セル２３０２の作成又は伝送の時刻と日付に基づき作成される。必要なＡＰＵ２３２６によってセルの実行に必要な最低数のＡＰＵが与えられる。サンドボックス・サイズ２３２８により、セルの実行に必要なＤＲＡＭと関連する必要なＡＰＵ内に、保護されたメモリ量が与えられる。前回のセルＩＤ２３３０により、シーケンシャルな実行を要求する１グループのセル（ストリーミング・データなど）内の前回のセルの識別子が提供される。

【００８２】

実行セクション２３３２にはセルのコア情報が含まれる。この情報には、ＤＭＡコマンド・リスト２３３４と、プログラム２３３６と、データ２３３８とが含まれる。プログラム２３３６には、ＡＰＵプログラム２３６０や２３３８などの、ＡＰＵによって実行されるプログラム（“アプレット”と呼ばれる）が含まれ、データ２３３８にはこれらのプログラムを用いて処理されるデータが含まれる。ＤＭＡコマンド・リスト２３３４には、プログラムの起動に必要な一連のＤＭＡコマンドが含まれる。これらのＤＭＡコマンドには、ＤＭＡコマンド２３４０、２３５０、２３５５、２３５８が含まれる。ＰＵはＤＭＡＣへこれらのＤＭＡコマンドを出す。

【００８３】

ＤＭＡコマンド２３４０にはＶＩＤ２３４２が含まれる。ＶＩＤ２３４２は、ＤＭＡコマンドが出されたときに、物理ＩＤに対して対応付けられるＡＰＵのバーチャルＩＤである。ＤＭＡコマンド２３４０には、ロード・コマンド２３４４とアドレス２３４６も含まれる。ロード・コマンド２３４４はＡＰＵにＤＲＡＭから特定の情報を読み出して、ローカル・ストレージの中へ入れるように命令する。アドレス２３４６によって、この特定情報を含むＤＲＡＭ内のバーチャル・アドレスが与えられる。この特定情報は、プログラム・セクション２３３６からのプログラム、データ・セクション２３３８からのデータ、又

はその他のデータなどであってよい。最終的に、ＤＭＡコマンド２３４０にはローカル・ストレージのアドレス２３４８が含まれる。このアドレスにより、情報をロードできそうなローカル・ストレージのアドレスが識別される。ＤＭＡコマンド２３５０には同様の情報が含まれる。その他のコマンドも使用可能である。

【００８４】

ＤＭＡコマンド・リスト２３３４には、一連のキック・コマンド（キック・コマンド２３５５や２３５８など）も含まれる。キック・コマンドとは、ＰＵによってＡＰＵへ出される、セルの処理を開始するコマンドである。ＤＭＡキック・コマンド２３５５にはバーチャル・ＡＰＵ　ＩＤ２３５２と、キック・コマンド２３５４と、プログラム・カウンタ２３５６とが含まれる。バーチャルＡＰＵ　ＩＤ２３５２はキックすべき対象ＡＰＵを識別し、キック・コマンド２３５４は関連するキック・コマンドを与え、プログラム・カウンタ２３５６はプログラムの実行用プログラム・カウンタのためのアドレスを与える。ＤＭＡキック・コマンド２３５８は同じＡＰＵ又は別のＡＰＵに対して同様の情報を与える。

【００８５】

上述のように、ＰＵは独立したプロセッサとしてＡＰＵを扱い、コプロセッサとして扱うものではない。したがって、ＡＰＵによる処理を制御するために、ＰＵは遠隔手順呼出しに類似したコマンドを使用する。これらのコマンドは“ＡＰＵ遠隔手順呼出し（ＡＲＰＣ）”と呼ばれる。ＰＵは一連のＤＭＡコマンドをＤＭＡＣへ出すことにより、ＡＲＰＣを実行する。ＤＭＡＣはＡＰＵプログラムとそれに関連するスタック・フレームとをＡＰＵのローカル・ストレージへロードする。次いでＰＵはＡＰＵへ最初のキックを出し、ＡＰＵプログラムを実行する。

【００８６】

図２４にアプレットを実行するためのＡＲＰＣのステップを例示する。指定のＡＰＵによるアプレットの処理の開始時にＰＵが実行するこれらのステップが、図２４の第１の部分２４０４に示されている。

【００８７】

ステップ２４１０で、ＰＵはアプレットを評価し、次にアプレットの処理用ＡＰＵを指定する。ステップ２４１２で、ＰＵは、必要な単複のサンドボックス用のメモリ・アクセス・キーの設定を行うＤＭＡコマンドをＤＭＡＣへ出すことにより、アプレットの実行用スペースをＤＲＡＭ内に割り振る。ステップ２４１４で、ＰＵは指定ＡＰＵへの割込み要求による、アプレットの完了信号の伝送を可能にする。ステップ２４１８で、ＰＵはＤＲＡＭからＡＰＵのローカル・ストレージへアプレットをロードするＤＭＡコマンドをＤＭＡＣへ出す。ステップ２４２０で、ＤＭＡコマンドが実行され、ＤＲＡＭからＡＰＵのローカル・ストレージへアプレットが読み出される。ステップ２４２２で、ＰＵは、アプレットと関連付けられたスタック・フレームをＤＲＡＭからＡＰＵのローカル・ストレージへロードするＤＭＡコマンドをＤＭＡＣへ出す。ステップ２４２３で、ＤＭＡコマンドが実行され、スタック・フレームがＤＲＡＭからＡＰＵのローカル・ストレージへ読み出される。ステップ２４２４で、ＰＵは、ＤＭＡＣがＡＰＵへキーを割り当てて、ステップ２４１２で指定された、一又は複数のハードウエア・サンドボックスからのデータ読み出しと、その一又は複数のハードウエア・サンドボックスへのデータ書き込みを行うことをＡＰＵに許可するＤＭＡコマンドを出す。ステップ２４２６で、ＤＭＡＣはＡＰＵへ割り当てられたキーを用いて、キー管理テーブル（ＫＴＡＢ）の更新を行う。ステップ２４２８で、ＰＵは、プログラムの処理を開始するＤＭＡコマンド“キック”をＡＰＵに出す。特定のアプレットに応じて、特定のＡＲＰＣの実行時に、ＰＵによって他のＤＭＡコマンドを出してもよい。

【００８８】

上述のように、図２４の第２の部分２４０４はアプレットの実行時にＡＰＵにより行われるステップを例示するものである。ステップ２４３０で、ＡＰＵは、ステップ２４２８で出されるキック・コマンドに応じてアプレットの実行を開始する。ステップ２４３２で

、アプレットの指示で、ＡＰＵはアプレットの関連スタック・フレームの評価を行う。ステップ２４３４で、ＡＰＵはＤＭＡＣへ複数のＤＭＡコマンドを出し、スタック・フレームが必要に応じてＤＲＡＭからＡＰＵのローカル・ストレージへ指定するデータのロードを行う。ステップ２４３６で、これらのＤＭＡコマンドが実行され、データはＤＲＡＭからＡＰＵのローカル・ストレージへ読み出される。ステップ２４３８で、ＡＰＵはアプレットを実行し、ある結果を出力する。ステップ２４４０で、ＡＰＵはＤＭＡＣへＤＭＡコマンドを出し、ＤＲＡＭにその結果を格納する。ステップ２４４２で、ＤＭＡコマンドが実行され、アプレットの結果がＡＰＵのローカル・ストレージからＤＲＡＭへ書き込まれる。ステップ２４４４で、ＡＰＵはＰＵへ割込み要求を出し、ＡＲＰＣが完了したことを示す信号伝送を行う。

【００８９】

ＰＵの指示の下で独立にタスクを実行するＡＰＵの能力によって、１グループのＡＰＵと、１グループのＡＰＵと関連付けられたメモリ・リソースとを拡張タスクの実行専用にすることが可能になる。例えば、１つのＰＵは、１以上のＡＰＵと、これらの１以上のＡＰＵと関連付けられた１グループのメモリ・サンドボックスとを、拡張された時間中、ネットワークを介して伝送されてくるデータの受信専用とし、また、１以上の他のＡＰＵとそれらと関連付けられたメモリ・サンドボックスへ、この時間中受信したデータの更なる処理を行うための送信専用とすることができる。この能力は、ネットワーク１０４を介して伝送されるストリーミングデータ（ストリーミングＭＰＥＧ又はストリーミングＡＴＲＡＣオーディオ又はビデオ・データなど）の処理によって特に好適である。ＰＵは、１以上のＡＰＵ及びそれらと関連付けられたメモリ・サンドボックスをこれらのデータの受信専用とし、１以上の他のＡＰＵ及びそれらと関連付けられたメモリ・サンドボックスをこれらのデータの解凍と処理専用とすることができる。言い換えれば、ＰＵはＡＰＵのグループとそれらと関連付けられたメモリ・サンドボックスとの間でこのようなデータ処理を行うための専用パイプライン関係の確立を行うことができる。

【００９０】

しかし、このような処理を効率的に実行するためには、パイプラインの専用ＡＰＵとメモリ・サンドボックスとが、データ・ストリームを含むアプレットの処理が行われない時間中もパイプライン専用のままであることが望ましい。言い換えれば、専用ＡＰＵとその関連付けられたサンドボックスが、これらの時間中、予約状態のままに置かれることが望ましい。アプレットの処理の完了時における、ＡＰＵとその関連付けられた一又は複数のメモリ・サンドボックスを予約、すなわちリザーブ状態としておくことは、“常駐終了”と呼ばれる。常駐終了はＰＵからの命令に応じて行われる。

【００９１】

図２５、２６Ａ及び２６Ｂは、１グループのＡＰＵ及びそれらと関連するサンドボックスを含む、ストリーミングデータ（ストリーミングＭＰＥＧデータなど）を処理するための専用パイプライン構造の設定を例示する。図２５に示すように、このパイプライン構造の構成要素にはＰＥ２５０２とＤＲＡＭ２５１８とが含まれる。ＰＥ２５０２にはＰＵ２５０４とＤＭＡＣ２５０６とが含まれ、複数のＡＰＵには、ＡＰＵ２５０８と、ＡＰＵ２５１０と、ＡＰＵ２５１２とが含まれる。広帯域バス２５１６によりＤＭＡＣ２５０６はＤＲＡＭ２５１８と接続される。ＤＲＡＭ２５１８には複数のサンドボックス（サンドボックス２５２０、サンドボックス２５２２、サンドボックス２５２４、サンドボックス２５２６など）が含まれる。

【００９２】

図２６Ａに専用パイプラインを設定するためのステップを例示する。ステップ２６１０で、ＰＵ２５０４はネットワーク・アプレットを処理するようにＡＰＵ２５０８を割り当てる。ネットワーク・アプレットはネットワーク１０４のネットワーク・プロトコルの処理用プログラムを有する。この場合、このプロトコルは、伝送制御プロトコル／インターネット用プロトコル（ＴＣＰ／ＩＰ）である。このプロトコルに従うＴＣＰ／ＩＰデータ・パケットは、ネットワーク１０４を介して伝送される。受信時に、ＡＰＵ２５０８はこ

れらのパケットを処理し、パケット内のデータを組み立て、ソフトウエア・セル１０２の中へ入れる。ステップ２６１２で、ＰＵ２５０４は、ネットワーク・アプレットの処理の完了時に常駐終了を実行するようにＡＰＵ２５０８に指示する。ステップ２６１４で、ＰＵ２５０４は、ＡＰＵ２５１０及び２５１２がＭＰＥＧアプレットの処理を行うように割り当てる。ステップ２６１５で、ＰＵ２５０４はＭＰＥＧアプレットの処理の完了時に常駐終了を実行するようにＡＰＵ２５１０と２５１２に指示する。ステップ２６１６で、ＰＵ２５０４は、ＡＰＵ２５０８とＡＰＵ２５１０によるアクセス用ソース・サンドボックスとしてサンドボックス２５２０を指定する。ステップ２６１８で、ＰＵ２５０４はＡＰＵ２５１０によるアクセス用宛先サンドボックスとしてサンドボックス２５２２を指定する。ステップ２６２０で、ＰＵ２５０４は、ＡＰＵ２５０８とＡＰＵ２５１２によるアクセス用ソース・サンドボックスとしてサンドボックス２５２４を指定する。ステップ２６２２で、ＰＵ２５０４は、ＡＰＵ２５１２によるアクセス用宛先サンドボックスとしてサンドボックス２５２６を指定する。ステップ２６２４で、ＡＰＵ２５１０とＡＰＵ２５１２とは、それぞれ、ソース・サンドボックス２５２０とソース・サンドボックス２５２４の範囲内のメモリ・ブロックへ同期読取りコマンドを送り、これらのメモリ・ブロックをブロッキング状態に設定する。最後に、処理はステップ２６２８へ移り、そこで専用のパイプラインの設定が完了し、パイプライン専用のリソースが予約される。このようにして、ＡＰＵ２５０８、２５１０、２５１２及びそれらと関連するサンドボックス２５２０、２５２２、２５２４、及び２５２６は予約状態に入る。

【００９３】

図２６Ｂにこの専用パイプラインによるストリーミングＭＰＥＧデータの処理ステップを例示する。ステップ２６３０で、ＡＰＵ２５０８はネットワーク・アプレットを処理し、そのローカル・ストレージの中でＴＣＰ／ＩＰデータパケットをネットワーク１０４から受信する。ステップ２６３２で、ＡＰＵ２５０８はこれらのＴＣＰ／ＩＰデータ・パケットを処理し、これらのパケット内のデータを組み立て、ソフトウエア・セル１０２の中へ入れる。ステップ２６３４で、ＡＰＵ２５０８はソフトウエア・セルのヘッダ２３２０（図２３）をチェックし、セルがＭＰＥＧデータを含むかどうかの判定を行う。セルがＭＰＥＧデータを含まない場合、ステップ２６３６で、ＡＰＵ２５０８は専用パイプラインに含まれない他のＡＰＵによって他のデータを処理するために、ＤＲＡＭ２５１８内に指定される汎用サンドボックスへセルを伝送する。ＡＰＵ２５０８はこの伝送についてＰＵ２５０４に通知する。

【００９４】

一方、ソフトウエア・セルがＭＰＥＧデータを含む場合は、ステップ２６３８で、ＡＰＵ２５０８はそのセルの前のセルＩＤ２３３０（図２３）をチェックし、そのセルが属するＭＰＥＧデータ・ストリームを識別する。ステップ２６４０で、ＡＰＵ２５０８はセル処理用の専用パイプラインのＡＰＵを選択する。この場合、ＡＰＵ２５０８はこれらのデータを処理するＡＰＵ２５１０を選択する。この選択は前回のセルＩＤ２３３０とロード・バランシング・ファクタ（負荷平衡係数）とに基づく。例えば、そのソフトウエア・セルが属するＭＰＥＧデータ・ストリームの前回のソフトウエア・セルが処理用としてＡＰＵ２５１０へ送られたことが前回のセルＩＤ２３３０によって示されている場合、現在のソフトウエア・セルも通常の処理用としてＡＰＵ２５１０へ送られる。ステップ２６４２で、ＡＰＵ２５０８は、サンドボックス２５２０へＭＰＥＧデータを書き込む、同期書き込みコマンドを出す。このサンドボックスはあらかじめブロッキング状態に設定されているので、ステップ２６４４で、ＭＰＥＧデータはサンドボックス２５２０からＡＰＵ２５１０のローカル・ストレージへ自動的に読み出される。ステップ２６４６で、ＡＰＵ２５１０はそのローカル・ストレージでＭＰＥＧデータを処理してビデオ・データを生成する。ステップ２６４８で、ＡＰＵ２５１０はサンドボックス２５２２へビデオ・データを書き込む。ステップ２６５０で、ＡＰＵ２５１０は同期読み出しコマンドをサンドボックス２５２０へ出し、このサンドボックスに追加ＭＰＥＧデータ受信の準備をする。ステップ２６５２で、ＡＰＵ２５１０は常駐完了処理を行う。この処理により、このＡＰＵは予約

状態に入り、この予約状態の間、ＡＰＵはＭＰＥＧデータ・ストリームの中で追加ＭＰＥＧデータの処理を行うべく待機する。

【００９５】

他のタイプのデータ処理用として、１グループのＡＰＵ及びそれらと関連するサンドボックス間でその他の専用構造の設定が可能である。例えば、図２７に示すように、ＡＰＵの専用グループ（ＡＰＵ２７０２、２７０８、２７１４など）を設定し、３次元オブジェクトに対して幾何学変換を実行して２次元ディスプレイ・リストの生成を行うことが可能となる。これらの２次元ディスプレイ・リストを他のＡＰＵによって更に処理（レンダーし）、画素データの生成を行うようにすることが可能である。この処理を実行するために、３次元オブジェクトと、これらのオブジェクト処理から結果として生じるディスプレイ・リストの格納用として、サンドボックスが、ＡＰＵ２７０２、２７０８、２４１４の専用となる。例えば、ソース・サンドボックス２７０４、２７１０、２７１６は、それぞれ、ＡＰＵ２７０２、２７０８、２７１４によって処理された３次元オブジェクトの格納専用となる。同様に、宛先サンドボックス２７０６、２７１２、２７１８はそれぞれ、ＡＰＵ２７０２、ＡＰＵ２７０８、ＡＰＵ２７１４によるこれらの３次元オブジェクトの処理から結果として生じるディスプレイ・リストの格納専用となる。

【００９６】

調整用ＡＰＵ２７２０は、そのローカル・ストレージにおける、宛先サンドボックス２７０６、２７１２、２７１８からのディスプレイ・リストの受信専用である。ＡＰＵ２７２０はこれらのディスプレイ・リスト間で調整を行い、画素データのレンダリングのためにこれらのディスプレイ・リストを他のＡＰＵへ送る。

【００９７】

システム１０１のプロセッサは絶対タイマーも使用する。この絶対タイマーはＡＰＵとＰＥの他のエレメントへクロック信号を出力する。このクロック信号はこれらのエレメントを駆動するクロック信号に依存せず、かつ、このクロック信号より高速である。この絶対タイマーの利用が図２８に例示されている。

【００９８】

この図に示すように、絶対タイマーによってＡＰＵによるタスク・パフォーマンスのためのタイム・バジェット（割り当て時間）が決定される。このタイム・バジェットによって、これらのタスクの完了時間が設定されるが、この時間はＡＰＵによるタスク処理に必要な時間より長い時間になる。その結果、各タスクについて、タイム・バジェットの範囲内に、ビジーな時間とスタンドバイ時間とが存在することになる。全てのアプレットは、ＡＰＵの実際の処理時間あるいは速度にかかわらず、このタイム・バジェットに基づいて処理を行うように書かれる。例えば、ＰＥの特定のＡＰＵ用として、タイム・バジェット２８０４のビジー時間２８０２中に特定のタスクを行うことができる。ビジー時間２８０２がタイム・バジェット２８０４未満であるため、スタンドバイ時間２８０６はタイム・バジェット中に生じる。このスタンドバイ時間中、ＡＰＵはＡＰＵが消費するパワーが少なくなるスリープモードに入る。

【００９９】

タイム・バジェット２８０４が満了するまで、他のＡＰＵ又はＰＥの他のエレメントがタスク処理の結果を予想することはしない。したがって、ＡＰＵの実際の処理速度にかかわらず、絶対タイマーによって決定されるタイム・バジェットを用いて、ＡＰＵの処理結果が常時調整される。

【０１００】

将来、ＡＰＵによる処理速度は更に高速になる。しかし、絶対タイマーによって設定されるタイム・バジェットは同じままである。例えば、図２８に示すように、将来のＡＰＵは更に短時間でタスクを実行することになり、したがって、スタンバイ時間は更に長くなるであろう。したがって、ビジー時間２８０８はビジー時間２８０２よりも短くなり、スタンバイ時間２８１０はスタンバイ時間２８０６よりも長くなる。しかし、絶対タイマーによって設定された同じタイム・バジェットに基づいて処理を行うようにプログラムが書

かれているので、ＡＰＵ間の処理結果の調整が維持される。その結果、更に高速のＡＰＵが、その処理の結果が予測される時点でコンフリクトを生じることなく、低速のＡＰＵ用として書かれたプログラムの処理を行うことが可能となる。

【０１０１】

動作速度の向上や動作速度が異なることに起因するＡＰＵの並列処理の調整問題に対しては、ＡＰＵ間での調整を決定する絶対タイマーに代えて、ＰＵまたは１以上の指定ＡＰＵにおいて、ＡＰＵが実行している特定の命令（マイクロコード）の分析をアプレットの処理時に行うようにすることもできる。“オペレーションなし”（“ＮＯＯＰ”）命令を命令の中へ挿入し、ＡＰＵのいくつかによってこの命令を実行してアプレットによって予測されるＡＰＵによる処理を１ステップずつ適切に行うことが可能となる。命令の中へこれらのＮＯＯＰを挿入することにより、全ての命令のＡＰＵによる実行を行うための正しいタイミングの維持が可能となる。

【０１０２】

上述のように、各処理エレメント（ＰＥ）には少なくとも１つの処理ユニット（ＰＵ）と、１つ以上の複数の付加処理デバイス（ＡＰＵ）が含まれ、ＰＵの調整及び制御下で、１つ以上のアプリケーションによりデータの並列処理がＡＰＵによって実行される。これに関連して、また、手短に図１８を参照すると、図１８にはＡＰＵと共用メモリとの間にデータをやりとりして送るための、例示的なデータレベル同期（ＤＬＳ）機構の状態が示されていた。しかし、前述のように、このＤＬＳ機構は外部デバイスへの、及び／又は外部デバイスからのデータの処理にも好適に利用される。更に、以下の説明からわかるように、Ｉ／Ｏデバイスを扱うためにＤＬＳを利用することにより、その他の利点が与えられる。例えば、現在のところそれらの利点としては、デバイス実装の詳細を隠す能力、複数のデバイスを扱う柔軟性、１つ以上の外部ソースからのストリーミング・データの処理を行うＡＰＵの能力、が挙げられる。図２９に留意すると、図２９には本発明の概念による例示的なＤＬＳアーキテクチャが示されている。

【０１０３】

図２９において、構成２９００にはＰＥ２９００と、ＤＲＡＭ（又は共用メモリ）２９１０と、インターフェースデバイス（インターフェース）２９１５と、外部デバイス２９２０とが含まれる。ＰＥ２９９０は１つ以上の処理エレメントを示し、さらにＰＥ２９００には、少なくとも１つのＰＵ２９０５と、１つ以上のＡＰＵ（ＡＰＵ２９０１、２９０２、２９０３により示す）とが含まれる。ＰＥ２９９０はバス２９０６を介して共用メモリと結合されており、またＰＥ２９９０は前述のＤＬＳ機構とメモリ保護機構を用いて、ＤＲＡＭ２９１０へのデータの書き込みと、ＤＲＡＭ２９１０からのデータの読み出しを行う。本発明の概念によると、現在のところ、このＤＬＳ機構は外部デバイスと用いるように拡張されている。そのために、また図２９からわかるように、ＰＥ２９９０はバス２９０７を介してインターフェース２９１５と結合されており、一方でＤＲＡＭ２９１０はバス２９０８を介してインターフェース２９１５と結合されている。またインターフェース２９１５は、バス２９１６を介して、図中で外部デバイス２９２０として示される１つ以上の外部デバイスと結合されている。外部デバイス２９２０は、ハードディスク又は光ディスクなどのメモリ、Ｉ／Ｏプロセッサ又は画像処理用プロセッサなどのプロセッサ、トランシーバ（ネットワークトランシーバなど）等を含む複数のデバイスのうちの任意の１つとすることができるが、外部デバイスはこれらに限定されるものではない。バス２９１６は外部デバイス２９２０をインターフェース２９１５と結合するための、１つ以上の通信バスを示し、それらは例えば、パケットネットワーク接続部、交換式ネットワーク接続部、シリアルバス、並列バス、などである。以下に更に説明しているように、バス２９０８はインターフェース２９１５を介して、ＤＲＡＭ２９１０へのアクセスを直接行う。

【０１０４】

図３０を参照すると、まず本発明の概念が、外部メモリ３０２０に関連して外部デバイスとして例示されている。特に、各ＡＰＵはバス２９０７を介して、インターフェース２９１５へ要求を送り、及び／又は、バス２９０７を介してインターフェース２９１５から

の応答を受け取る、１つ以上のチャネルと関連付けられている。これについては図３０に例示されており、ＡＰＵ２９０１に対し、チャネル０（要求）（２９３０）とチャネル１（応答）（２９３５）により関連付けられている。説明のため、エレメント２９３０と２９３５の双方とも、１つ以上のＦＩＦＯ（first-in-first out）バッファである。簡素化のため、他のＡＰＵにおける、順次転送される要求及び応答の他のチャネルは、図３０の破線矢印で示す。ここでは、エレメント２９３０と２９３５にはコマンドと状態情報だけが格納されるとする。要求チャネルと応答チャネルはバス２９０７を介して伝送コントローラ(transfer controller)２９４５を用いてコマンドと状態情報を交換する。説明のため、伝送コントローラ２９４５はＡＰＵからの要求の処理を行う外部ＤＭＡＣである。図３０からわかるように、インターフェース２９１５には保護テーブル２９４０が含まれる。これは、ＰＵ２９０５の内部ＤＭＡＣ（図示せず）において図１９と関連して上述した、キー管理テーブル１９０２に追加するものである。外部保護テーブル（保護テーブル２９４０など）の利用により、外部メモリ３０２０から、ＤＲＡＭ２９１０の１つ以上のサンドボックスへの直接的データ・ストリーミングが改善される。説明のため、保護テーブル２９４０の値は、上述の方法と同様の方法でＰＵにより割り当てられる。保護テーブル２９４０を利用して、伝送コントローラ２９４５は、ＤＲＡＭ２９１０の１つ、又は複数のサンドボックスと外部メモリ３０２０との間で、いずれかの方向にデータ通信を行うために、伝送バス２９５０を制御する。外部保護テーブルは必要とされない点に留意されたい。しかし、外部保護テーブルがない場合、好適にはＰＵから、例えば、ＰＵと関連付けられて同様に設けられた要求／応答チャネルから、適切な許可が送られるのが望ましい。上述のように、ＰＵは“信頼できるプログラム(trusted program)”を実行するため、関連の読み出し／書き込み許可の割り当てがなされるべきである。更に、保護テーブル２９４０が拡張され、外部デバイスへのアクセス、又はその１部へのアクセスを制御することもできる。これについては外部メモリ３０２０の特定のメモリ・アドレス領域により示される。

【０１０５】

引き続き図３０を参照し、また図３１と３２を特に参照すると、外部メモリ３０２０からＡＰＵ２９０１のローカル・メモリ（図示せず）へのデータ伝送に利用するＤＬＳ法が例示されている。この方法は図２４について上述した方法の変形であるので、全てのステップを以下に繰り返さず、また図３１には全てのステップを示していない。この例では、ＡＰＵ２９０１によってアプレット（前述の、ソフトウエアの種類）が実行されると仮定する。このＡＰＵは外部メモリ３０２０に現在格納されているデータ上で動作する必要があるものである。まず図３１のステップ３１０５を見ると、ＰＵ２９０５はＡＰＵ２９０１によって用いられるＤＲＡＭ２９１０のメモリの割り当てを行う。更にステップ３１１０で、ＰＵ２９０５はバス２９０７を介して、割り当てられたメモリに対する適切なキー保護値をインターフェース２９１５の伝送コントローラ２９４５へ与え、ＤＲＡＭ２９１０のメモリへのアクセス制御を行う保護テーブル２９４０で用いる。ＰＵはＡＰＵ２９０１によってアクセス制御が行われるように、そのキー管理テーブルの更新を行うことに加え、ＰＵは上記処理を、図１９に関連して上述した方法で行う。ステップ３１１５で、ＰＵ２９０５はＡＰＵ２９０１によりアプレットの実行を開始する。このアプレットは現在のところ外部メモリ３０２０に格納されているデータへのアクセスを要求する。ＡＰＵ２９０１を参照すると、ステップ３１３０で、このＡＰＵにより、チャネル０を介してインターフェース２９１５へ要求が送られ、外部メモリ３０２０などの外部デバイスから、ＤＲＡＭ２９１０へのデータ伝送が行われる。この要求は、ＤＲＡＭ２９１０への同期書き込みオペレーションに対するものであり、またこの要求は前述のＤＭＡコマンドと形式が類似している。説明のため、この要求には、例えば外部デバイス（ここでは、外部メモリ３０２０）や、利用される外部デバイスの一部（ここでは、外部メモリ３０２０のデータ位置）の識別子、伝送量、ＤＲＡＭ２９１０の対応アドレス、及び、オペレーションの種類が、読み出しオペレーションか書き込みオペレーションか（ここでは、ＤＲＡＭ２９１０への同期書き込みオペレーション）、が挙げられる。デバイスや利用されるデバイスの

一部などの外部デバイス情報やサイズ情報は、例えばアプレット関連情報として与えられる。また、他の種類の外部デバイス・パラメータには、デバイス番号やポート番号、仮想チャネル識別子などが挙げられるが、これらに限定されるものでないことに留意されたい。

【０１０６】

図３２、及び引き続き図３０を参照すると、インターフェース２９１５の伝送コントローラ２９１５により、ステップ３２０５において、ＡＰＵ２９０１からの伝送要求が受け取られる。ステップ３２１０で、伝送コントローラ２９４５により保護テーブル２９４０に格納された値のチェックが行われ、ＤＲＡＭ２９１０で要求されたメモリの値域が有効であるかが判断される。メモリの値域が無効である場合、ステップ３２２０で、伝送コントローラ２９４５により、チャネル１を介した伝送が拒否される。しかし、メモリの値域が有効である場合、ステップ３２２５で、伝送コントローラにより伝送が開始され、完了するまで伝送バス２９５０を介して、ＤＭＡ様式でデータ伝送が継続して行われる（ステップ３２３０）。伝送が完了すると、伝送コントローラ２９４５はステップ３２３５で、応答チャネル１を介して伝送完了メッセージを送る。外部デバイス情報も保護テーブル２９４０に格納される場合は、ステップ３２１０や３２１５などで、ＡＰＵ２９０１が伝送を進める前に、その外部デバイスが利用許可を有しているかどうかのチェックも行う。

【０１０７】

本発明によれば、外部メモリ３０２０からＡＰＵ２９０１へのデータ伝送は、図１８に関連して上述しているように、ＤＬＳを用いて行われる。これに関連して、図３３にはＤＬＳの用途が例示されており、図３３ではデータ・フロー（破線矢印により示す）が示されている。同期書き込みオペレーションはインターフェース２９１５を通じてＤＲＡＭ２９１０へ行われ、一方で同期読み出しオペレーションはＡＰＵ２９０１を通じて行われ、ローカル・メモリ２９８１への格納用に、ＤＲＡＭ２９１０からデータの読み出しが行われる。図１７Ｂから１７Ｄに関連して上述しているように、ＤＲＡＭ２９１０による個々の同期書き込みオペレーションと同期読み出しオペレーション処理とが、共用メモリのエンプティ状態、ブロッキング状態、及び、フル状態によって制御される（例えば、関連する状態情報のＬＳアドレスフィールド、ＡＰＵ　ＩＤ、及びＦ／Ｅビット値、により示す）。したがって、例えば、最初にＰＵがＡＰＵによって用いられるＤＲＡＭへのデータ伝送を行うといったことを必要とせずに、外部デバイスからＡＰＵのローカル・メモリへ、効率的に、つまり低オーバーヘッドでデータの伝送を行うことが可能である。換言すれば、ＰＵで実行されるべきＩ／Ｏ処理が、安全かつ同期して、ＡＰＵにより実行されるアーキテクチャレベルにまで、その実行レベルが引き下げられている。これにより、複数のストリーミングアプリケーションにおいて、ＡＰＵが、外部メモリの複数のバッファからのデータを素早く読出し、動作することが可能となる。

【０１０８】

同期読み出しオペレーション、つまり、ＤＲＡＭ２９１０から外部デバイス（外部メモリ３０２０など）への伝送、も上述と同様の方法で可能であることに留意されたい。これについては、図３４に例示されており、また引き続き図３１、３２のフローチャートを参照する。例えば、ステップ３１１５で、ＰＵ２９０５は、アプレットのＡＰＵ２９０１により実行を開始する。このアプレットは、外部メモリ３０２０に格納するためにデータの伝送を必要とする。そのため、同期読み出しオペレーションに対する要求が、チャネル０を介してＡＰＵによってインターフェース２９１５へ送られる（つまり、ＤＲＡＭ２９１０から外部デバイス（外部メモリ３０２０）へのデータ伝送）。この要求は前述のＤＭＡコマンドと同様の形式である。この要求には、例えば外部デバイス（ここでは、外部メモリ３０２０）や、利用される外部デバイスの一部（ここでは、外部メモリ３０２０のデータ位置）の識別子、伝送量、ＤＲＡＭ２９１０の対応アドレス、及び、外部デバイスオペレーションが、読み出しオペレーションか書き込みオペレーションか（ここでは、ＤＲＡＭ２９１０からの同期読み出しオペレーション）、が挙げられる。

【０１０９】

これに関連して、ＤＬＳの利用により、インターフェース２９１５が同期方式でＤＲＡＭ２９１０からのデータの読み出すのを可能にし、一方で、ＡＰＵ２９０１はＤＲＡＭ２９１０への書き込みを行う。これについては図３４に示されており、ＡＰＵ２９０１のローカル・メモリ２９８１から外部デバイスへのデータ・フロー（破線矢印により示す）が図３４に示されている。これに関連して、インターフェース２９１５がデータの読み出しを試みたかどうかに関わらず、ＤＬＳの様々な状態がＦ／Ｅビット値により示されている。このため、ＤＲＡＭ２９１５の関連メモリ・ロケーションに対する上述の状態情報が、図３５に示すように変更される。この状態情報には、上述のようにＦ／Ｅビット３５０１と、デバイスＩＤ３５０２と、データが伝送される伝送先であるアドレス３５０３とが含まれる。デバイスＩＤ３５０２により、ＡＰＵなどのデバイスのタイプが識別され、又はインターフェースなどの別のデバイスが識別される。これに関連して、インターフェース２９１５にはすでに保護テーブルを持つＤＭＡタイプのコントローラが含まれているので、アドレス・フィールドは要求され得ない点に留意すべきである。つまり、ＤＲＡＭ２９１０は、同期読み出しオペレーションが許可された場合に、インターフェース２９１５（デバイスＩＤ３５０２により識別される）へ単に通知を出すだけである。

【０１１０】

上述のように、図３０により示されている構成は単に例示的なものであり、その他のタイプの外部デバイスへ直接的に拡張することができる。例えば、図３６を検討する。この図は図３０と同じものである。このため同じ参照符号は同様の要素を表しており、そのような要素についてはここではさらに説明しない。図３６では、外部デバイスは物理レシーバ（physical receiver：ＰＨＹ）３６２０によって示される。物理レシーバ３６２０はパス３６２１を介して、イーサネットなどのネットワーク（図示せず）からパケット通信を受信し、受信したビット・ストリームのデコーディングを行い、パケットのストリームを与える。この例では、ＡＰＵによって実行されたアプレットは受信したパケットストリームのリンク層処理を示す。つまり、ＡＰＵ２９０１はＰＨＹ３６２０からのデータの処理を行うために上述のＤＬＳ機構を利用して、Ｉ部のプロトコルスタック（ＴＣＰ／ＩＰ（通信制御プロトコル／インターネットプロトコル）など）に対し処理を行う。

【０１１１】

引き続き図３６を参照し、また特に図３７を参照すると、例示的なハンドシェーキング・シーケンスが示されており、外部デバイスを用いたデータレベル同期の利用がさらに例示されている。この図では、共有メモリ（ＤＲＡＭ２９１０など）の対応部分がエンプティ状態で開始されると仮定する。最初に、ＡＰＵ（ＡＰＵ２９０１など）が要求チャネルを介して、外部デバイスから共用メモリの識別部分への同期書き込みオペレーションを要求するコマンドを、インターフェース（インターフェース２９１５など）へ送る。ここでは、パスインターフェース２９１５のチェックを行う保護テーブルが、外部デバイスを用いて伝送を開始すると仮定する。一方で、ＡＰＵは共用メモリのこれらの部分からの同期読み出しオペレーションの実行を試みる。インターフェースによって書き込みが行われているデータはまだないので、共用メモリの対応部分がブロッキング状態に入る。この状態で、前述のように、ＡＰＵに対するローカル・メモリのアドレスが共用メモリのこれらの部分に対し、関連する状態情報フィールドに格納される。続いて、インターフェースが共用メモリの識別部分への同期書き込みオペレーションを行った後、ＡＰＵに対し、同期読み出しオペレーションが、格納されたアドレス値により示されるアドレスへ行われる。その結果、共用メモリのこれらの部分がエンプティ状態へ戻る。その他の順列に対する状態変化（図１８に示す）を示す、同様のハンドシェーキング・シーケンスは簡明であり、ここには説明していない。

【０１１２】

１つ以上のＡＰＵからの要求は、インターフェースを通じて、各要求の完了を待つ必要がないことがわかる。例えば、ＡＰＵは１つ以上の外部デバイスからのデータに対し、インターフェースへ複数の要求を出すことができる。ＤＬＳの利用により、（インターフェースを介して）ＡＰＵ、あるいは外部デバイスのいずれから、同期読み出しオペレーショ

ンが行われるまで、データが確実に共用メモリに保護される。そのため、ＤＬＳの利用により、ＡＰＵと１つ以上の外部デバイスとの間に実行されるバースト伝送と並列バースト伝送とが可能になる。

【０１１３】

説明のため、図３６に示すデバイスを用いた本発明の概念の他の適用例を図３８に示す。ＡＰＵ２９０１はインターフェース２９１５を介して、多数の外部デバイスと接続している。これらの外部デバイスは、ＵＳＢ（ユニバーサル・シリアル・バス）デバイス、ＩＥＥＥ１３９４（一般に“ファイアワイヤ”と呼ばれる）デバイス、シリアル・デバイス、ハードディスク、ネットワーク・デバイス（ネットワークトランシーバなど）、光ディスク、である。その他の外部デバイス（図示せず）としては、Ｉ／Ｏプロセッサ、又は画像処理用プロセッサなどのプロセッサが挙げられる。ＡＰＵ２９０１は１つ以上のこれらの外部デバイスからのデータの読み出しを行うために、要求４３０１を送る。そのため、各デバイスはインターフェース２９１５の制御を介して、続いてＤＲＡＭ２９１０のそれぞれに割り当てられたメモリ・ロケーションへ同期書き込みオペレーションを行う。ＤＲＡＭ２９１０のメモリ・ロケーションは、メモリ・ロケーション４３１１、４３１２、４３１３、４３１４、４３１５、４３１６により示される（各メモリ・ロケーションには、関連付けられた状態情報が含まれることも示し、それらは単にＦ／Ｅビットにより示される）。ＡＰＵ２９０１は、同期読み出しオペレーションを行うことにより、バス２９０６を介して、ＤＲＡＭ２９１０からデータを読出す。

【０１１４】

図３９を参照すると、本発明の概念の別の実施形態が例示されている。図３９に簡略化したストリーミング構成３７００を示す。構成３７００には、ＰＥ３７９０と、ＤＲＡＭ（又は共用メモリ）３７１０と、インターフェースデバイス（インターフェース）３７１５と、物理トランシーバ（ＰＨＹ）により示される外部デバイスとが含まれる。外部デバイス３７２０はインターフェース３７１５の一部ともでき、例えば、インターフェース３７１５と結合することもできる。ＰＥ３７９０は１つ以上の処理エレメントを示し、更にＰＥ３７９０には少なくとも１つのＰＵ３７０５と、１つ以上のＡＰＵ（ＡＰＵ３７０１、３７０２、３７０３により示す）とが含まれる。ＰＥ３７９０はバス３７０６を介して共用メモリと結合されている。ＰＥ３７９０はまた、前述のＤＬＳ機構とメモリ保護機構を用いて、ＤＲＡＭ３７１０へのデータの書き込みと、ＤＲＡＭ３７１０からのデータの読み出しを行う。本発明の概念によれば、現在のところ、ＤＬＳ機構は外部デバイスと用いるように拡張されている。そのために、また、図３９からわかるように、各ＡＰＵは、バス３７０７を介してインターフェース３７１５と結合されている。特に各ＡＰＵは、１つ以上のチャネルと関連付けられており、インターフェース３７１５への要求の送信、及び／又は、インターフェース３７１５からの応答の受信、及び／又は、インターフェース３７１５とのデータ通信が、バス３７０７を介して行われる。これについては図３９に例示されており、ＡＰＵ３７０１に対して、チャネル０（入力）（３７３０）と、チャネル１（出力）（３７３５）が関連付けられている。説明のため、エレメント３７３０と３７３５の双方が、１つ、またはそれ以上のＦＩＦＯバッファである。その他のＡＰＵに関しては、バス３７０７を介して転送されるそれらの入力及び出力のチャネルは、図３９に破線矢印により示される。

【０１１５】

インターフェース３７１５には、チャネル・インターフェース・エレメント３７４５と３７５０とが含まれ、インターフェース３７１５により、ＡＰＵ３７０１とＰＨＹ３７２０との間のそれぞれの入出力データストリームが結合される。説明のため、チャネル・インターフェース・エレメント３７５０はトランシーバであり、（ａ）ＡＰＵ３７０１の出力チャネルから送られるデータを、バス３７０７と３７１６を介してエンコードし、バス３７２１を介して伝送する、及び（ｂ）バス３７２１から受け取ったデータをデコードし、バス３７１６と３７０７を介して、ＡＰＵ３７０１の入力チャネルへデコードしたデータを送る。バス３７２１は外部デバイス３７２０を、パケットネットワーク接続部、交換

式ネットワーク接続部、シリアルバス、並列バス、などへ結合するための、１つ以上の通信バスを示す。

【０１１６】

上述のように、インターフェース３７１５には、チャネル・インターフェース・エレメント３７４５と３７５０が含まれる。チャネル・インターフェース・エレメント３７４５はＡＰＵ３７０１から出力チャネルを受信する。本発明によれば、出力チャネル自体には制御チャネルとデータチャネルとが含まれる。この制御チャネルにより、上述の要求と応答のチャネルが形成され、一方で、データのチャネルにより、外部デバイスへ送られるデータが転送される。この例では、要求チャネルにより外部デバイス識別子（ここではＰＨＹ３７２０）を含むコマンド情報や、実行されるオペレーション（ここでは、書き込みオペレーション）などが転送される。出力チャネルのデータチャネル部は、ＰＨＹ３７２０によってエンコードされ、バス３７２１を介して伝送されるデータを示す。同様に、チャネル・インターフェース・エレメント３７４５は、ＡＰＵ３７０１の入力チャネルへ結合される。入力チャネル自体には、制御チャネルとデータチャネルとが含まれる。制御チャネルにより、上述の要求と応答のチャネルが形成され、一方で、ＡＰＵへ送られるデータはデータチャネルによって転送される。この例では、外部デバイス識別子（ここでは、ＰＨＹ３７２０）や、実行されるオペレーション（ここでは、読み出しオペレーション）などを含むコマンド情報が、要求チャネルによって転送される。この入力チャネルのデータチャネル部により、バス３７２１から受信した信号受信の結果として、ＰＨＹ３７２０によってデコードされたデータが示される。

【０１１７】

この実施形態では、インターフェース３７１５内には保護テーブルが必要とされないことがわかる。特に、また更に以下に説明しているように、図１８に例示している前述のＤＬＳ機構とメモリ保護機構により、各ＡＰＵはＤＲＡＭ３７１０からのデータの読み出しと、ＤＲＡＭ３７１０へのデータの書き込みとが行われる。

【０１１８】

引き続き図３９を参照すると、外部デバイスからＡＰＵ２９０１のローカル・メモリ（図示せず）へのデータ伝送に用いる例示的な方法が図４０に示されている。この方法により、上述した図２４の方法も実行されると仮定する。その方法とは、ＰＵによる保護テーブル値の割り当て、ＡＰＵ３７０１などへのアプレットの割り当て、などである。説明のため、この例では、ＡＰＵ３７０１によって実行されるアプレットは、受信したパケットストリームのリンク層処理と、送信したパケットストリームに対するリンク層処理とを示す。つまり、ＡＰＵ２９０１によって上述のＤＬＳ機構を利用したプロトコルスタック（ＴＣＰ／ＩＰなど）の一部分に対する処理が行われる。

【０１１９】

図４０のステップ３８３０をまず参照すると、ＡＰＵ３７０１は、バス３７０７を介して、ＰＨＹ３７２０からの、また、ＰＨＹ３７２０へのデータ伝送要求を、インターフェース３７１５へ行う。この例では、入力チャネルと出力チャネルの双方のコマンド部を介してこの要求が行われる。しかし、入力チャネル及び出力チャネルの双方の利用は必要とされない。この要求には、外部デバイス（ここではＰＨＹ３７２０）の識別子などが含まれ、また、利用する外部デバイスの一部（特定のポートなど）や、外部デバイスのオペレーションの種類（ここでは、入力チャネルに対する外部デバイスからの読み出しと、出力チャネルに対する外部デバイスへの書き込み）も含まれる。デバイスや利用するデバイスの一部などの外部デバイス情報は、アプレット関連の情報などに与えられるとする。さらに、他の種類の外部デバイスパラメータには、デバイス番号やポート番号、仮想チャネル識別子、などを挙げられるが、これらに限定されるものではない。

【０１２０】

ステップ３８０５で、インターフェース３７１５は要求を受け取り、ＰＨＹ３７２０への、またＰＨＹ３７２０からの伝送を開始する。特定のハンドシェーキングがＰＨＹ３７２０に必要とされる範囲で、このハンドシェーキングがインターフェース３７１５によっ

て行われ、また、これがＡＰＵ３７０１にとってトランスパレントなものであるとする。ステップ３７１０で、インターフェース３７１５は伝送を開始し、伝送が完了するまで、ＰＨＹ３７２０からのデータ伝送を継続する（ステップ３８１５）。このようなデータ伝送は、例えば遠方終端への接続が切れるまでは継続される。この、ＰＨＹ３７２０への、またＰＨＹ３７２０からのデータ伝送の部分は、外部デバイスの種類に応じて、同期して、又は非同期に行うことができる。データは入力チャネルのデータチャネル部内に、例えば、エレメント３７３５のＦＩＦＯの一部などへ、伝送される。伝送が完了すると、インターフェース３７１５はステップ３８２０で、入出力チャネルのコマンド部の応答チャネルを介して、伝送完了メッセージを送る。

【０１２１】

本発明によれば、ＡＰＵ３７０１による、外部デバイス３７２０からの、また、外部デバイス３７２０へのデータ伝送は、ＤＬＳを用いて行われる。このＤＬＳの状態図は先述の図１８に説明している。これに関連して、ＤＬＳの用途を図４１に例示する。ＡＰＵと外部デバイスとの間のデータ・フローが破線矢印により示されている。図４１からわかるように、ＡＰＵ３７０１はバス３７０７による入力チャネルを介して、上述しているようにローカル・メモリの中にデータを直接受け取る。同様に、ＡＰＵ３７０１はバス３７０７による出力チャネルを介して、ローカル・メモリから直接データを送る。これにより、速くて低いオーバーヘッドの通信チャネルがもたらされる。例えば、ＰＨＹ３７２０を介して送信されるデータは、まずＤＲＡＭ３７１０に格納され得る。ＡＰＵ３７０１は同期読み出しオペレーションを介して、ＤＬＳを用いて、このデータをＤＲＡＭ３７１０から抽出する。これについては図４１に対応の破線矢印により例示している。次いで、ＡＰＵ３７０１はリンク層のアプレットによりこのデータの処理を行い、処理したデータ（フォーマットされたパケットのストリームなど）を（インターフェース３７１５を介して）ＰＨＹ３７２０へ送り、更にエンコードを行い、伝送する。同様に、ＰＨＹ３７２０によって受け取られたデータのデコードが行われ、データはローカル・メモリ３７８１に格納するために入力チャネルを介してＡＰＵ３７０１へ送られ、ＡＰＵ３７０１によって処理される。ＡＰＵ３７０１はリンク層のアプレット（ヘッダの除去、エラーチェックなど）によりデータの処理を行い、同期書き込みオペレーションを介して、ＤＬＳを用いて、処理されたデータをＤＲＡＭ３７１０へ送る。これについては図４１に、対応の破線矢印で描かれている。

【０１２２】

図４２に図３９に関連して示され、上記に説明されている、簡略化したストリーミング構成の変形を例示する。この実施形態は図３９に示す方法と同様の方法で動作するため、全ての図については詳細に説明していない。図４２の構成４０００にはＰＥ４０９０と、ＤＲＡＭ（又は、共用メモリ）４０１０と、インターフェース４０１５と、物理トランシーバ（ＰＨＹ）４０２０と物理トランシーバ（ＰＨＹ）４０８０により示される外部デバイス、とが含まれる。図４２からわかるように、説明のため、インターフェース４０１５は外部デバイス４０２０及び４０８０と結合されている。ＰＥ４０９０は１つ以上の処理エレメントを示し、更に、ＰＥ４０９０には少なくとも１つのＰＵ４００５と、１つ以上のＡＰＵ（ＡＰＵ４００１、４００２、４００３）とが含まれる。ＰＥ４０９０はバス４００６を介して共用メモリと結合されており、前述のＤＬＳとメモリ保護機構を用いて、ＤＲＡＭ４０１０へのデータの書き込みと、ＤＲＡＭ４０１０からのデータの読み出しとを行う。図４２からわかるように、ＡＰＵはバス４００７を介して、インターフェース４０１５と結合されている。特に、各ＡＰＵは１つ以上のチャネルと関連付けられており、バス４００７を介して、インターフェース４０１５への要求の送信、及び／又は、インターフェース４０１５からの応答の受信、及び／又はインターフェース４０１５とのデータ通信を行う。これについては図４２に例示されており、ＡＰＵ４００１に対し、入力／出力チャネル４０５０と入力／出力チャネル４０５５が関連付けられている。説明のため、これらのエレメントは双方とも、１つ以上のＦＩＦＯバッファである。バス４００７を介して形成されるその他のＡＰＵ用のその他の入力及び出力のチャネルは、図４２の破線矢

印により示されている。

【０１２３】

インターフェース４０１５にはエレメント４０３０と４０３５とが含まれ、各エレメントにはそれぞれ、２つのチャネル・インターフェース・エレメント４０３１、４０３２と４０３６、４０３７とが含まれる。（これらは図３９で説明している種類である。）これらのエレメントにより、データストリームがそれぞれの外部デバイス（ここでは、４０２０と４０８０）へ結合される。説明のため、この外部デバイスは、それぞれのバス４０２１と４０８１への、及び、それぞれのバス４０２１と４０８１からのデータ通信用トランシーバである。この外部デバイス４０２０、及び／又は４０８０には、例えばハードディスク又は光ディスクなどのメモリ、Ｉ／Ｏプロセッサ又は画像処理用プロセッサなどのプロセッサ、トランシーバ（ネットワークトランシーバなど）なども挙げられる。

【０１２４】

上述の通り、この例では、ＡＰＵ４００１は２つの入力／出力チャネル（４０５０と４０５５）へのアクセスを持ち、各チャネルには更に、コマンド情報とデータを転送する、上述の入出力チャネルが含まれる。上述した方法と同様の方法で、ＡＰＵ４００１によってアプレットが実行され、外部デバイス（４０２０と４０８０）との間にデータが転送される。ＡＰＵ４００１は複数の入力／出力チャネルへのアクセスを持つので、ＡＰＵ４００１がＤＲＡＭ４０１０へアクセスすることは要求されない。そのために、図４２に示す変形は、外部デバイス間にデータを転送する、効率的で高速な構成を提供する。

【０１２５】

このように、上記の内容は、単に本発明の原則を例示するものに過ぎず、当業者は、ここには明示的に記載されていないが、本発明の趣旨及び範囲から逸脱することなくその原則を具体化する数多くの代替の構成を考案しうるであろうことを理解されよう。例えば、本発明の概念が、データレベルの同期化を行う入力／出力インターフェースを採り上げて説明されているが、インターフェースが必ずしも入力と出力の双方を行う必要はなく、例えば、インターフェースは入力デバイスのみ、又は、出力デバイスのみ、又は、一般の入力／出力ネットワークで用いられる、入力デバイスと出力デバイスの任意の組合せ、などのうちの１つでありうる。

【図面の簡単な説明】

【０１２６】

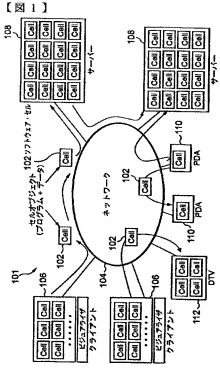
【図１】本発明によるコンピュータ・ネットワークのアーキテクチャ全体を示す説明図。
【図２】本発明によるプロセッサ・エレメント（ＰＥ）の構造を示す説明図。
【図３】本発明による広帯域エンジン（ＢＥ）の構造を示す説明図。
【図４】本発明による付加処理デバイス（ＡＰＵ）の構造を示す説明図。
【図５】本発明によるプロセッサ・エレメントと、ビジュアライザ（ＶＳ）と、光インターフェースとの構造を示す説明図。
【図６】本発明によるプロセッサ・エレメントの１つの組合せを示す説明図。
【図７】本発明によるプロセッサ・エレメントの別の組合せを示す説明図。
【図８】本発明によるプロセッサ・エレメントの更に別の組合せを示す説明図。
【図９】本発明によるプロセッサ・エレメントの更に別の組合せを示す説明図。
【図１０】本発明によるプロセッサ・エレメントの更に別の組合せを示す説明図。
【図１１Ａ】本発明によるチップ・パッケージ内へ光インターフェースを統合した例を示す説明図。
【図１１Ｂ】図１１Ａの光インターフェースを用いるプロセッサの１つの構成を示す説明図。
【図１１Ｃ】図１１Ａの光インターフェースを用いるプロセッサの別の構成を示す説明図。
【図１２Ａ】本発明によるメモリ・システムの構造を示す説明図。
【図１２Ｂ】本発明による第１の広帯域エンジンから第２の広帯域エンジンへのデータの書き込みを示す説明図。

【図１３】本発明によるプロセッサ・エレメント用の共用メモリの構造を示す説明図。
【図１４Ａ】図１３に示すメモリ・バンクの一構造を示す説明図。
【図１４Ｂ】図１３に示すメモリ・バンクの別の構造を示す説明図。
【図１５】本発明によるＤＭＡＣの構造を示す説明図。
【図１６】本発明によるＤＭＡＣの代替の構造を示す説明図。
【図１７Ａ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｂ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｃ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｄ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｅ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｆ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｇ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｈ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｉ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｊ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｋ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｌ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｍ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｎ】本発明によるデータ同期オペレーションを示す説明図。
【図１７Ｏ】本発明によるデータ同期オペレーションを示す説明図。
【図１８】本発明によるデータ同期方式によるメモリ・ロケーションの様々な状態を示す説明図。
【図１９】本発明によるハードウエア・サンドボックス用のキー管理テーブルの構造を示す説明図。
【図２０】本発明によるハードウエア・サンドボックス用のメモリ・アクセス・キーの格納方式を示す説明図。
【図２１】本発明によるハードウエア・サンドボックス用のメモリ・アクセス管理テーブルの構造を示す説明図。
【図２２】図１９のキー管理テーブルと、図２１のメモリ・アクセス制御テーブルとを用いてメモリ・サンドボックスにアクセスするステップを示すフローチャート。
【図２３】本発明によるソフトウエア・セルの構造を示す説明図。
【図２４】本発明による、ＡＰＵへ遠隔処理命令を出すステップを示すフローチャート。
【図２５】本発明によるストリーミング・データ処理専用パイプラインの構造を示す説明図。
【図２６Ａ】本発明によるストリーミング・データの処理時の図２５の専用パイプラインによって実行されるステップを示すフローチャート。
【図２６Ｂ】本発明によるストリーミング・データの処理時の図２５の専用パイプラインによって実行されるステップを示すフローチャート。
【図２７】本発明によるストリーミング・データ処理専用パイプラインの他の構造を示す説明図。
【図２８】本発明によるＡ˜ＰＵによるアプリケーションとデータの並列処理を調整するための絶対タイマー方式を示す説明図。
【図２９】本発明の原則に従って、データレベル同期を用いた構成の実施形態を例示的に示す説明図。
【図３０】本発明の原則に従って、データレベル同期を用いた構成の実施形態を例示的に示す説明図。
【図３１】本発明の原則に従って、データレベル同期実行時に用いるフローチャートを例示的に示す説明図。
【図３２】本発明の原則に従って、データレベル同期実行時に用いるフローチャートを例示的に示す説明図。

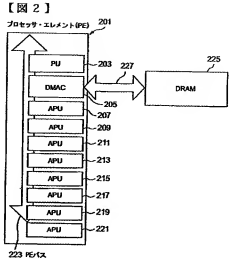
【図３３】ＡＰＵと外部デバイスとの間にデータレベル同期の利用を示す説明図。
【図３４】ＡＰＵと外部デバイスとの間にデータレベル同期の利用を示す説明図。
【図３５】本発明の原則に従ってデータレベル同期状態情報を例示的に示す説明図。
【図３６】本発明の原則に従ってデータレベル同期を用いた構成のもう１つの実記形態を例示的示す説明図。
【図３７】ハンドシェーキングシーケンスを例示的に示す説明図。
【図３８】本発明の原則に従って、データレベル同期を用いた構造用の別の実施形態を例示的に示す説明図。
【図３９】本発明の原則に従い、データレベル同期実行時に用いるフローチャートを例示的に示す図。
【図４０】本発明の原則に従い、データレベル同期の実行に用いる別のフローチャートを例示的に示す説明図。
【図４１】ＡＰＵと外部デバイスとの間に別のデータレベル同期の用途を例示的示す説明図。
【図４２】本発明の原則に従いデータレベル同期を用いた別の構成用の実施形態を例示的に示す説明図。
【符号の説明】
【０１２７】
１０１　システム
１０１０　キー
１０２　セル
１０４　ネットワーク
１０６　クライアント
１０８　サーバー・コンピュータ
１１０４　光インターフェース
１１０８　バス
１１１８，１１２２　ポート
１１２６　光導波路
１１６０，１１６２，１１６４，１１６６，１１８２，１１８４，１１８６，１１８８，
１１９０　光インターフェース
１２０６，１２３４，１２４２　コントロール
１２１２，１２４０　ユニット
１２２１　クロスバ交換機
１２３２　外部ポート
１２４４，１４１４，１４１６　バンク
１４０６　ブロック
１５０４　ノード
１６０７，１６０８　バス
１７２２　制御回路
１７２４，１７４２　制御論理回路
１７２６　ストレージ
１７２８，１７３１，１７３２，１７４６，１７５０　ロケーション
１７２９，１７５２，１７６０，１７６２　セグメント
１８８０　エンプティ状態
１８８２　フル状態
１８８４　ブロッキング状態
１９０２　キー管理テーブル
１９０６　キー
１９０８　マスク
２００６　格納位置

２００８，２０１０　セグメント
２０１２　キー
２１０２　アクセス管理テーブル
２１０６　アドレス
２１１０　キー
２１１０　キー・マスク
２２３　バス
２２７　高帯域メモリ接続部
２３０２，２３２０　セル
２３０８　ヘッダ
２３２２　インターフェース
２３３２　実行セクション
２３３４　リスト
２５２０，２５２２，２５２４，２５２６，２７０４　サンドボックス
２７０６　宛先サンドボックス
２９００　処理環境
２９０１，２９０２，２９０３　ＡＰＵ
２９０５　ＰＵ
２９０６～２９０８　バス
２９１０　ＤＲＡＭ
２９１５　インターフェース
２９２０　外部デバイス
２９３０　エレメント
２９４０　保護テーブル
２９４５　伝送コントローラ
２９５０　伝送バス
２９８１　メモリ
３０２０　外部メモリ
３６２０　物理レシーバ
３７０６，３７０７　バス
３７１５　インターフェース
３７２０　外部デバイス
３７３０，３７３５，３７４５，３７４０　エレメント
３７８１　メモリ
４０１５　インターフェース
４０２０　外部デバイス
４０３０，４０３１　エレメント
４０５０，４０５５　出力チャネル

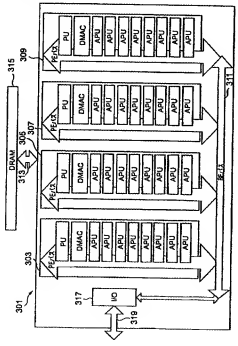
【図１】


108
サーバー
Cell
102 ソフトウェア・セル
ネットワーク
102
110
PDA
101
106
104
ビジュアライザ
クライアント
112
DTV

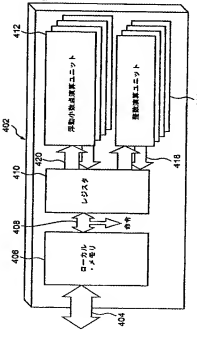
【図２】


プロセッサ・エレメント(PE)
201
PU
203
DMAC
205
APU
207
209
211
213
215
217
219
221
223 PEバス
227
225
DRAM

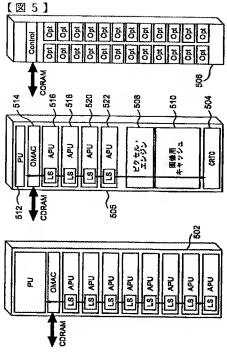
【図３】


315
309
311
DRAM
305
307
313
PE
PU
DMAC
APU
303
301
317
IO
319

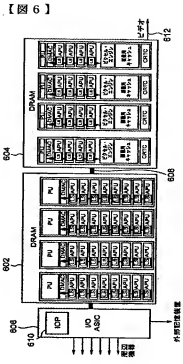
【図４】


412
浮動小数点演算ユニット
整数演算ユニット
414
402
410
420
418
レジスタ
406
408
ローカル・メモリ
404

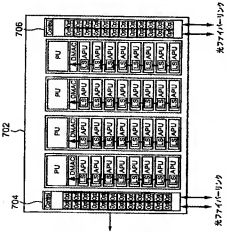
【図５】


Control
Opt
CDRAM
514
516
518
520
522
508
510
504
PU
DMAC
APU
LS
ビジュアライザエンジン
画像用キャッシュ
CRTC
512
606
502
CDRAM

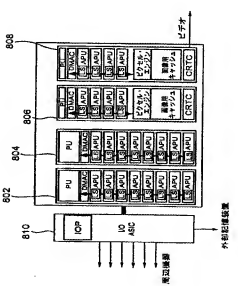
【図６】


ビデオ
612
DRAM
604
606
602
PU
DMAC
APU
LS
CRTC
608
610
IOP
IO
ASIC
外部記憶装置
周辺機器

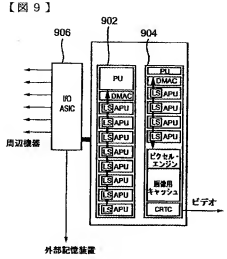
【図７】


706
光ファイバーリンク
702
704
PU
DMAC
APU
LS
外部記憶装置

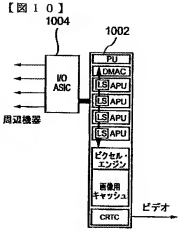
【図８】


ビデオ
808
806
804
802
810
PU
DMAC
APU
LS
CRTC
IOP
IO
ASIC
外部記憶装置
周辺機器

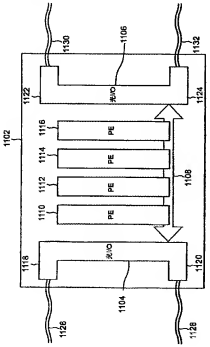
【図９】



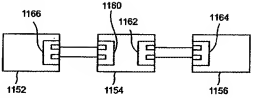
【図１０】



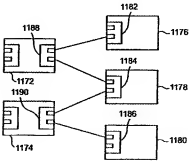
【図１１Ａ】



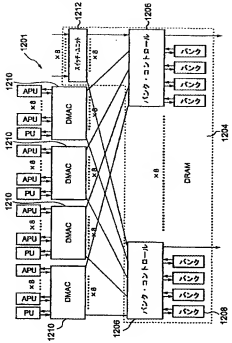
【図１１Ｂ】



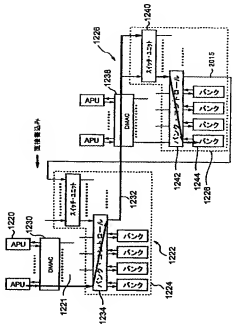
【図１１Ｃ】



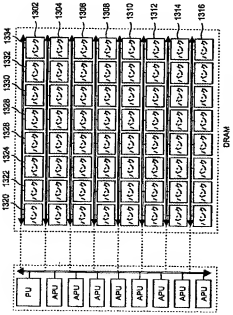
【図１２Ａ】



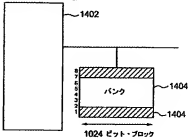
【図１２Ｂ】



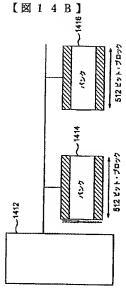
【図１３】



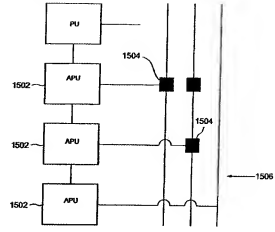
【図１４Ａ】



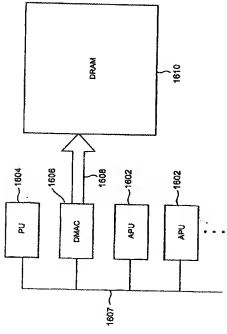
【図１４Ｂ】



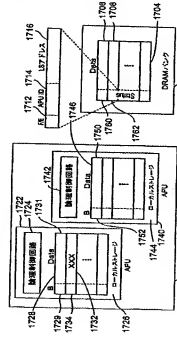
【図１５】



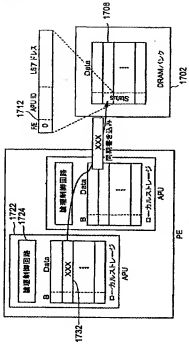
【図１６】



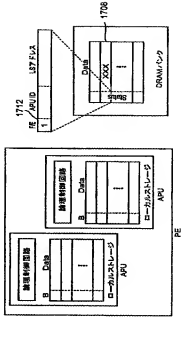
【図１７Ａ】



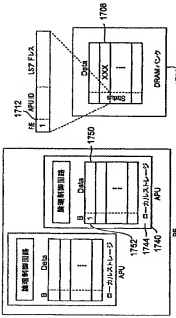
【図１７Ｂ】



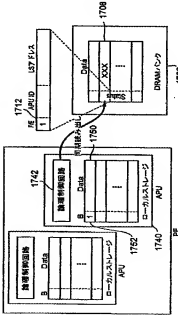
【図１７Ｃ】



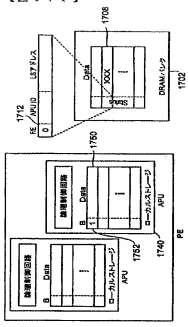
【図１７Ｄ】



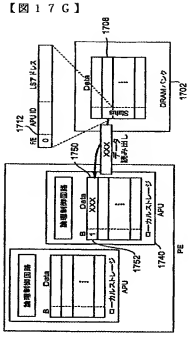
【図１７Ｅ】



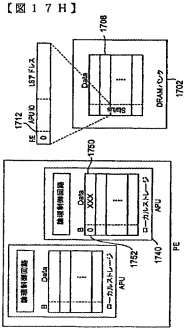
【図１７Ｆ】



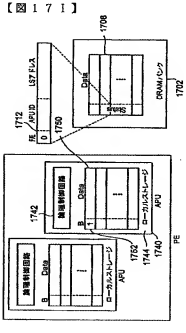
【図１７Ｇ】



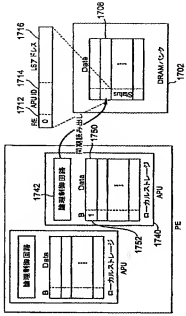
【図１７Ｈ】



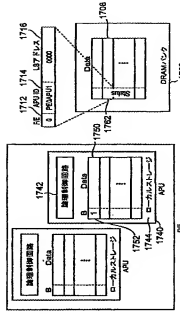
【図１７Ｉ】



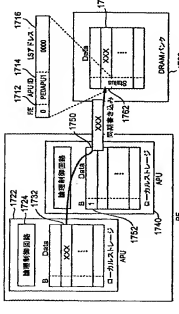
【図１７Ｊ】



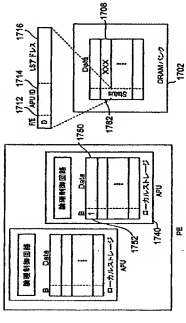
【図１７Ｋ】



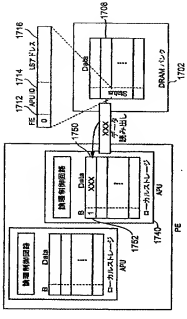
【図１７Ｌ】



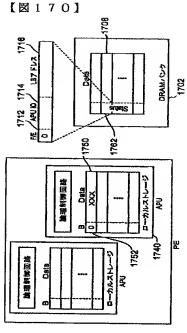
【図１７Ｍ】



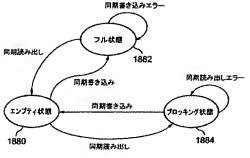
【図１７Ｎ】



【図１７Ｏ】



【図１８】

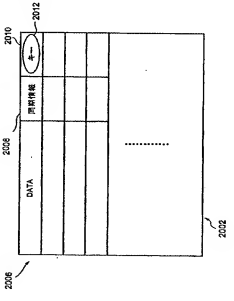


【図１９】

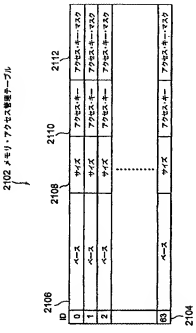
キー管理テーブル 1902

1904 ID	1906	1908
0	APUキー	キー・マスク
1	APUキー	キー・マスク
2	APUキー	キー・マスク
：	：	：
7	APUキー	キー・マスク

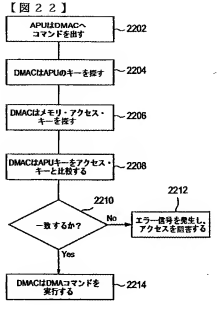
【図２０】



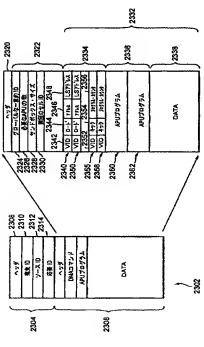
【図２１】



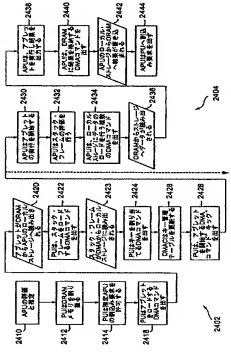
【図２２】



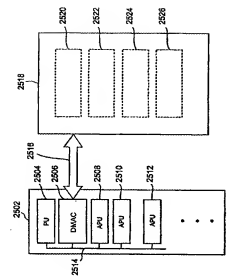
【図２３】



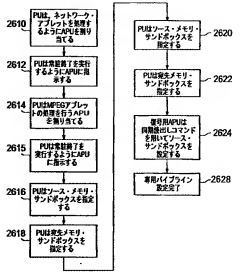
【図２４】



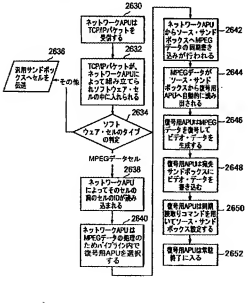
【図２５】



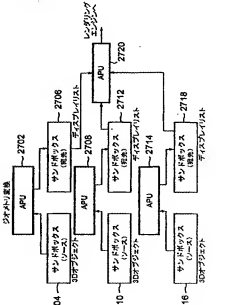
【図２６Ａ】



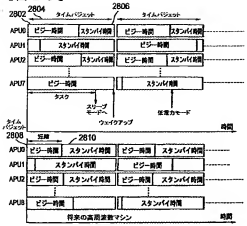
【図２６Ｂ】



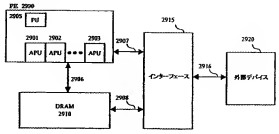
【図２７】



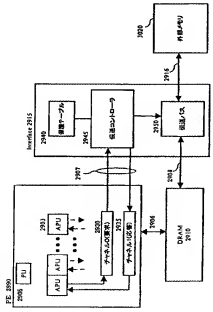
【図２８】



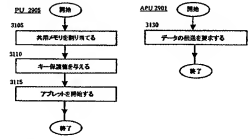
【図２９】



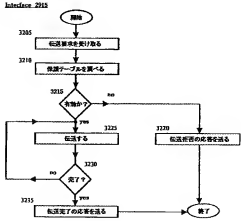
【図３０】



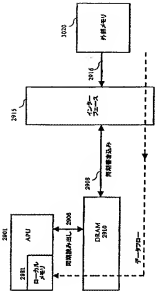
【図３１】



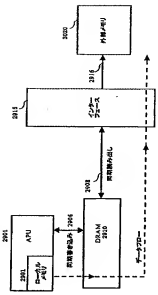
【図３２】



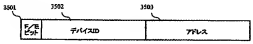
【図３３】



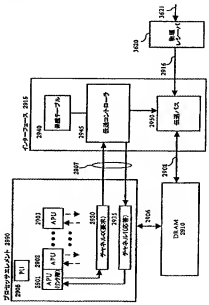
【図３４】



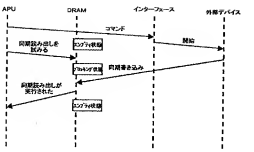
【図３５】



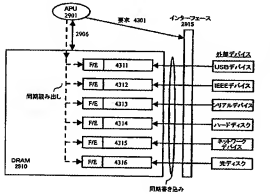
【図３６】



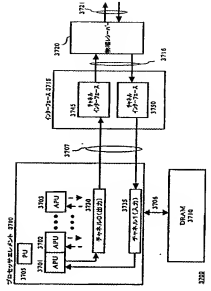
【図３７】



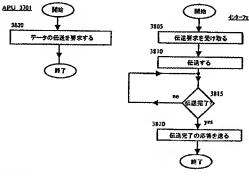
【図３８】



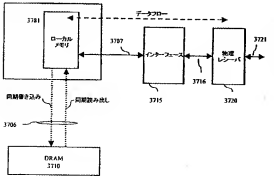
【図３９】



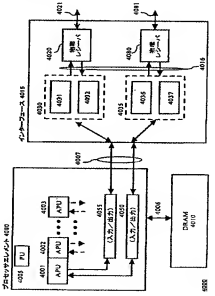
【図４０】



【図４１】



【図４２】



フロントページの続き

(72)発明者　山崎　剛
東京都港区南青山二丁目６番２１号　株式会社ソニー・コンピュータエンタテインメント内